



TimePunch

TimePunch API Specification 3.4

User Manual

Document Information:

| | |
|-----------------|--|
| Document-Name | User Manual, TimePunch API Specification 3.4 |
| Revision-Number | 25 |
| Saved at | 11.02.2019 |
| Number of Pages | 167 |

Publisher:

TimePunch KG
Bauhofstr. 34
68623 Lampertheim

Table of contents

| | |
|---|----|
| Introduction | 12 |
| Installation | 12 |
| Access..... | 12 |
| Service-Formats | 13 |
| Services | 14 |
| TimePunch Authentication Service | 14 |
| IsDatabaseValid | 14 |
| ValidateAuthentication | 15 |
| ValidateAuthenticationWithProductLine | 15 |
| GetCoveredUserProfiles | 16 |
| GetEntitledUserProfiles | 16 |
| GetLicensedUserProfiles | 17 |
| SearchAuthorizedUsers | 17 |
| SearchUserLogons | 18 |
| SetPassword..... | 18 |
| GetAutoRegistration | 19 |
| SaveAutoRegistration | 19 |
| TimePunch Configuration Service | 20 |
| LoadCostCenterSettings | 20 |
| SaveCostCenterSettings | 20 |
| LoadDatevExportSettings..... | 21 |
| SaveDatevExportSettings..... | 21 |
| LoadLexwareExportSettings | 21 |
| SaveLexwareExportSettings..... | 22 |
| LoadUserSettings | 22 |
| SaveUserSettings | 23 |
| TimePunch Common Task Service | 24 |
| GetActiveCommonTasks | 24 |
| GetAllCommonTasks | 24 |
| ImportTasks | 25 |
| LoadCommonTask | 25 |
| SaveCommonTasks | 26 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|----------------------------------|----|
| TimePunch Customer Service | 27 |
| CreateCustomer | 27 |
| SaveCustomers..... | 27 |
| ImportCustomers | 28 |
| LoadCustomers | 28 |
| MarkCustomersAsDeleted | 29 |
| DeleteCustomers | 29 |
| SearchCustomers | 30 |
| SearchActiveCustomers..... | 30 |
| TimePunch Holiday Service..... | 31 |
| GetCountries | 31 |
| GetPublicHolidays | 31 |
| SavePublicHolidays..... | 32 |
| ExportPublicHolidays | 32 |
| ImportPublicHolidays | 33 |
| TimePunch Licensing Service..... | 34 |
| AddLicense | 34 |
| GetApplicationLicenses..... | 34 |
| GetLicensingState | 35 |
| IsLicenseUpgradeable..... | 36 |
| RequestUpgradeOffer..... | 36 |
| GetUserLicenses | 37 |
| RemoveLicenses | 37 |
| RegisterUsers | 38 |
| UnregisterUsers..... | 38 |
| TimePunch Logging Service..... | 39 |
| GetActiveTimeEntry | 39 |
| SearchActiveTimeEntries..... | 39 |
| StartWorkTimeLogging..... | 40 |
| StartBreakTimeLogging..... | 41 |
| EndLogging | 41 |
| CancelLogging..... | 42 |
| LockSession..... | 42 |
| UnlockSession | 43 |
| SaveLoggingSettings..... | 43 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|--|----|
| SearchLoggingSettings | 44 |
| GetLoggingContext | 44 |
| ValidateAndSaveActiveTimeEntry | 45 |
| ValidateAndSaveRestrictedActiveTimeEntry | 45 |
| TimePunch Mailing Service | 46 |
| SendMail | 46 |
| SendMailWithAttachments | 47 |
| TimePunch Project Service | 48 |
| CreateProject | 48 |
| ExportProjectUserRelations | 49 |
| GetProjectUsers | 49 |
| GetTotalProjectTimes | 50 |
| ImportProjects | 50 |
| ImportProjectTasks | 51 |
| ImportProjectUserRelations | 51 |
| LoadProject | 52 |
| LoadProjectUserRelations | 52 |
| MarkProjectsAsDone | 53 |
| SaveProject | 53 |
| SaveProjectUserRelations | 53 |
| SearchAuthorizedProjects | 54 |
| SearchProjects | 54 |
| SetProjectLifeCycle | 55 |
| TimePunch Reporting Service | 56 |
| GetDailySummaries | 56 |
| GetMonthlySummaries | 56 |
| GetYearlySummaries | 57 |
| GetDailySummariesAnonymized | 58 |
| GetReportTimeEntriesUserBased | 58 |
| GetReportTimeEntriesProjectBased | 59 |
| GetReportProjectTimes | 59 |
| TimePunch Summary Service | 60 |
| GetSummaryLocks | 60 |
| LoadAllYearlySummaries | 60 |
| LoadDailySummaries | 61 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|---|----|
| LoadMonthlySummary | 61 |
| LoadTimeAccounts | 62 |
| LoadYearlySummary | 62 |
| ReCalculateAllSummaries | 63 |
| SaveMonthlySummary..... | 63 |
| SaveYearlySummary..... | 64 |
| SetSummaryLocks | 64 |
| TimePunch Sync Service | 65 |
| GetUseableSyncUserProfiles | 65 |
| SearchTimeEntriesForSynchronisation | 66 |
| SearchProjectsForSynchronisation | 66 |
| UpdateLastSyncDate | 67 |
| ApplyProjects | 67 |
| ApplyTimeEntries..... | 68 |
| TimePunch Async Sync Service..... | 69 |
| ApplyProjects | 69 |
| ApplyTimeEntries..... | 69 |
| ITpSyncCallbackService..... | 70 |
| TimePunch TimeEntry Service | 71 |
| AssignTimeEntriesToCustomer..... | 71 |
| AssignTimeEntriesToProjects | 71 |
| BatchFilterUpdate | 72 |
| CheckTimeEntryOverlapping..... | 72 |
| CopyTimeEntries | 73 |
| CreateBreakTimeDummy | 73 |
| CreateNewTimeEntry | 74 |
| DeleteTimeEntries | 74 |
| DeleteTimeEntryFilter | 75 |
| ImportTimeEntries | 75 |
| LoadTimeEntries..... | 76 |
| LoadTimeEntryFilter | 76 |
| MarkTimeEntries..... | 77 |
| SaveTimeEntryFilter | 77 |
| SearchTimeEntries | 78 |
| SearchTimeEntryFilters..... | 78 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|--|----|
| SearchAssignableUsers..... | 79 |
| SearchCoveredUserGroups | 79 |
| UnmarkTimeEntries | 80 |
| UpdateTimeEntryFilter | 80 |
| ValidateAndSaveBreakReplacement | 81 |
| ValidateAndSaveRestrictedTimeEntry | 81 |
| ValidateAndSaveTimeEntry | 82 |
| ValidateAndSaveTimeEntrySeries | 82 |
| ValidateAndSaveTimeEntryCopies | 83 |
| TimePunch User Group Service | 84 |
| CreateUserGroup | 84 |
| DeleteUserGroups | 84 |
| ImportUserGroupAssignments | 85 |
| LoadUserGroup | 85 |
| SaveUserGroup..... | 86 |
| SaveUserGroupAssignments | 86 |
| SearchAssignableUsers..... | 87 |
| SearchCoveredUserGroups | 87 |
| SearchUserGroupAssignments | 88 |
| SearchUserGroup | 88 |
| TimePunch User Profile Service..... | 89 |
| CreateNewUserProfile | 89 |
| DeleteUserProfileDataOlderThan | 90 |
| DownloadProfileImages..... | 90 |
| ExportUserProfileData | 91 |
| ImportUserProfileData..... | 91 |
| LoadUserProfileDefaults..... | 92 |
| LoadUserProfileDetails | 92 |
| LoadUserProfileRights | 93 |
| LoadUserProfileSensitive | 93 |
| SaveUserProfileDefaults | 94 |
| SaveUserProfileDetails..... | 94 |
| SaveUserProfileRights..... | 95 |
| SaveUserProfileSensitive | 95 |
| SearchUserProfilesToDelete | 96 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|---|-----|
| UploadProfileImage | 96 |
| TimePunch Work Model Service | 97 |
| AddEmptyWorkmodel | 97 |
| GetWorkDayModels | 97 |
| GetWorkModel | 98 |
| GetWorkModels | 98 |
| SaveWorkModels | 99 |
| Entities | 100 |
| Core | 100 |
| TpFault | 100 |
| TpAuthenticationFault : TpFault | 100 |
| PagingContextDto | 101 |
| TimePunch Authentication Service | 102 |
| TpAuthentication | 102 |
| UserProfileDto | 102 |
| UserLogonDto : UserProfileDto | 103 |
| UserSearchDto : PagingContextDto | 103 |
| AutoregistrationDto | 103 |
| TimePunch Configuration Service | 104 |
| CostCenterDto | 104 |
| SalaryExportDto | 104 |
| DatevExportSaveDto: SalaryExportDto | 105 |
| DatevExportDto: DatevExportSaveDto | 105 |
| LexwareExportDto: SalaryExportDto | 106 |
| In addition to the actual wage information, this class also contains data on the set cost centers | 106 |
| UserSettingsDto | 106 |
| TimePunch Customer Service | 107 |
| CustomerDto | 107 |
| CustomerSearchDto | 108 |
| TimePunch Holiday Service | 110 |
| CountryDto | 110 |
| RegionDto | 110 |
| PublicHolidayDto | 110 |
| TimePunch Licensing Service | 112 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|---|-----|
| ApplicationLicenseDto | 112 |
| UserLicenseDto | 112 |
| TimePunch Logging Service..... | 113 |
| ActiveTimeEntryDto..... | 113 |
| ActiveTimeEntrySearchDto..... | 114 |
| LoggingContextDto | 114 |
| LoggingSettingsDto | 115 |
| TimePunch Mailing Service | 116 |
| PlainMailAddressDto | 116 |
| UserMailAddressDto | 116 |
| MailAttachementDto | 116 |
| TimePunch Project Service | 117 |
| ProjectDto..... | 117 |
| ProjectSearchDto..... | 119 |
| ProjectTaskDto : TaskDto | 120 |
| ProjectTimeDto..... | 120 |
| ProjectUserDto : UserProfileDto | 120 |
| ProjectUserRelationSaveDto : UserProfileDto..... | 121 |
| ProjectUserRelationDto : ProjectUserRelationSaveDto | 121 |
| TaskDto | 121 |
| TimePunch Reporting Service | 123 |
| ReportCustomerDto | 123 |
| ReportDailySummaryDto | 124 |
| ReportDSGVOInfoDto..... | 125 |
| ReportLeadingUserDto..... | 126 |
| ReportProjectDto..... | 126 |
| ReportProjectTimeDto..... | 129 |
| ReportProjectUserDto | 130 |
| ReportSummaryDto | 130 |
| ReportTaskDto | 133 |
| ReportTimeEntryDto | 133 |
| ReportUserDto | 136 |
| ReportWorkdayDto | 140 |
| ReportWorkModelDto | 141 |
| ReportYearlySummaryDto..... | 142 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|--|-----|
| TimePunch Summary Service | 144 |
| DailySummaryDto..... | 144 |
| MonthlySummarySaveDto..... | 145 |
| MonthlySummaryDto : MonthlySummarySaveDto | 146 |
| SummaryLockDto | 147 |
| TimeAccountDto | 147 |
| YearlySummarySaveDto | 148 |
| YearlySummaryDto : YearlySummarySaveDto | 148 |
| TimePunch Sync Service | 150 |
| SyncUserProfileDto..... | 150 |
| TimePunch TimeEntry Service | 151 |
| TimeEntryDto : TimeEntrySaveDto | 151 |
| TimeEntryFilterSearchDto..... | 152 |
| TimeEntryRestrictedSaveDto | 152 |
| TimeEntryResultDto..... | 153 |
| TimeEntrySaveDto..... | 153 |
| TimeEntrySearchDto : PagingContextDto | 154 |
| TimeEntrySeriesDto | 156 |
| TimePunch User Group Service | 157 |
| UserGroupDto | 157 |
| UserGroupMemberDto : UserGrouSaveMemberDto | 157 |
| UserGrouSaveMemberDto | 157 |
| UserGroupSearchDto : PagingContextDto..... | 158 |
| TimePunch User Profile Service..... | 159 |
| UserProfileDataDto..... | 159 |
| UserProfileDefaultsDto : UserProfileDto | 159 |
| UserProfileDetailsDto : UserProfileDto..... | 160 |
| UserProfileImageDto : UserProfileDto..... | 162 |
| UserProfileRightsDto: UserProfileDto..... | 162 |
| UserProfileSensitiveDto : UserProfileDto | 162 |
| TimePunch Work Model Service..... | 164 |
| WorkModelDto | 164 |
| WorkdayDto | 164 |
| Constant values | 165 |
| Core | 165 |

TIMEPUNCH API SPECIFICATION 3.4

| | |
|---|-----|
| ApplicationKeys..... | 165 |
| Annex..... | 166 |
| Hash the password with the MD5 cryptography provider..... | 166 |

TIMEPUNCH API SPECIFICATION 3.4

Introduction

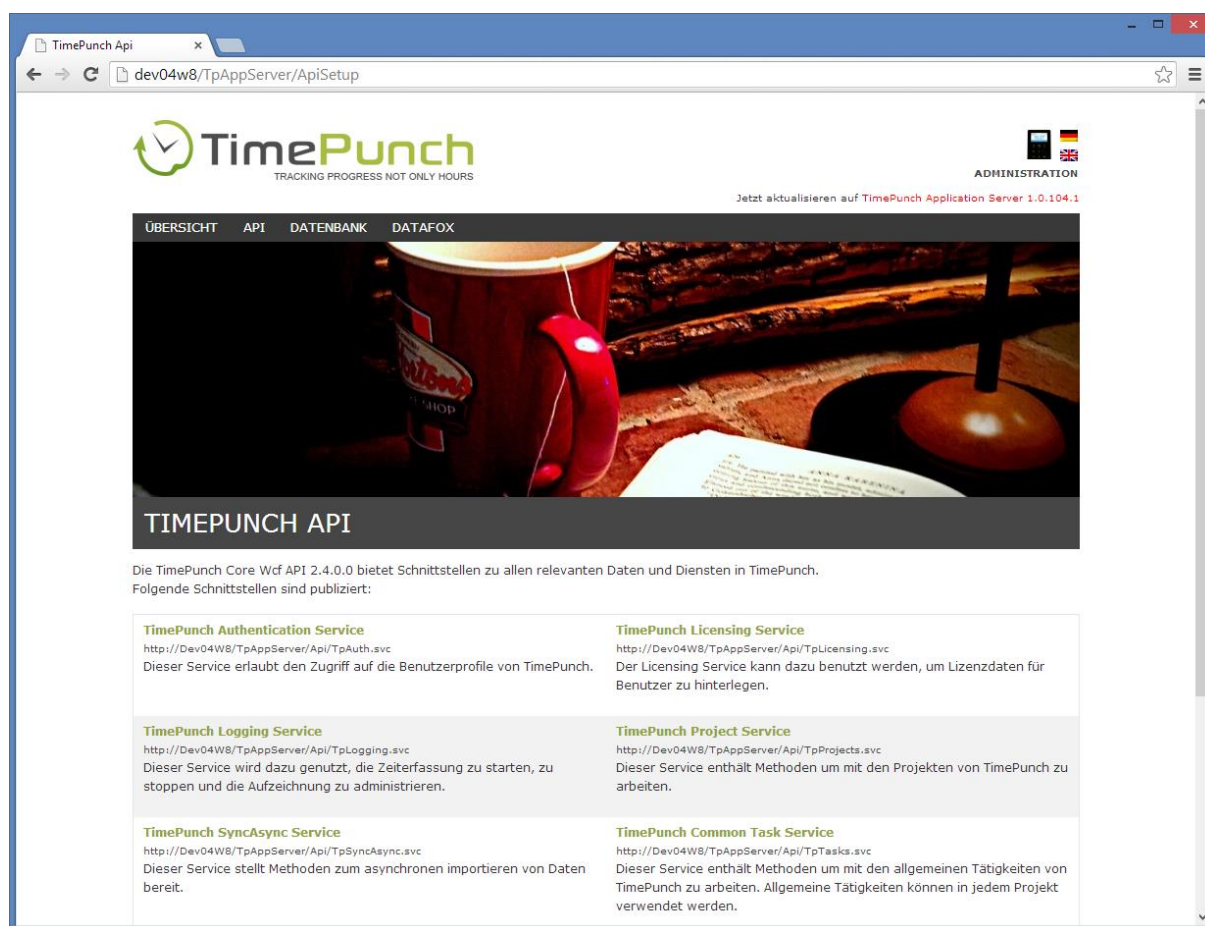
This document describes the usage of the TimePunch API. It's the goal to enable the developer to communicate with the backend of TimePunch, to exchange data and to integrate TimePunch into the IT-Infrastructure.

Installation

The TimePunch API will be installed and delivered with the TimePunch Application Server.

Access

Within the Server the exported services can be accessed with a click to the menu "API". The API page contains the URLs to the exported services.



TimePunch
TRACKING PROGRESS NOT ONLY HOURS

ADMINISTRATION
Jetzt aktualisieren auf TimePunch Application Server 1.0.104.1

ÜBERSICHT API DATENBANK DATAFOX

TIMEPUNCH API

Die TimePunch Core Wcf API 2.4.0.0 bietet Schnittstellen zu allen relevanten Daten und Diensten in TimePunch.
Folgende Schnittstellen sind publiziert:

| | |
|---|---|
| TimePunch Authentication Service http://Dev04W8/TpAppServer/Api/TPAuth.svc Dieser Service erlaubt den Zugriff auf die Benutzerprofile von TimePunch. | TimePunch Licensing Service http://Dev04W8/TpAppServer/Api/TPLicensing.svc Der Licensing Service kann dazu benutzt werden, um Lizenzdaten für Benutzer zu hinterlegen. |
| TimePunch Logging Service http://Dev04W8/TpAppServer/Api/TPLogging.svc Dieser Service wird dazu genutzt, die Zeiterfassung zu starten, zu stoppen und die Aufzeichnung zu administrieren. | TimePunch Project Service http://Dev04W8/TpAppServer/Api/TPProjects.svc Dieser Service enthält Methoden um mit den Projekten von TimePunch zu arbeiten. |
| TimePunch SyncAsync Service http://Dev04W8/TpAppServer/Api/TPSyncAsync.svc Dieser Service stellt Methoden zum asynchronen importieren von Daten bereit. | TimePunch Common Task Service http://Dev04W8/TpAppServer/Api/TPTasks.svc Dieser Service enthält Methoden um mit den allgemeinen Tätigkeiten von TimePunch zu arbeiten. Allgemeine Tätigkeiten können in jedem Projekt verwendet werden. |

Service-Formats

The TimePunch API has been implemented with the Windows Communication Foundation (WCF). The Services can be accessed via the WS and SOAP Binding.

A Web API is planned, but yet not available.

WS-http Binding:

The WS-http Binding URL equals the address that is listed in the TimePunch Server. E.g. z.B. <http://tpServer/TimePunch/API/TpAuth.svc>

Basic-http Binding (SOAP Binding):

The address for the Basic-http Binding, or SOAP Binding, equals the address of the WS-http binding with a SOAP postfix. e.g. <http://tpServer/TimePunch/API/TpAuth.svc/soap>

TIMEPUNCH API SPECIFICATION 3.4

Services

The following pages describe the different services of the TimePunch API.

TimePunch Authentication Service

This service enables the access to the user profiles of TimePunch.

IsDatabaseValid

This method checks the database structure of the transferred modules. If the database structure of a module does not fit, an error message is returned in the fault object.

```
void IsDatabaseValid(  
    TpAuthentication authentication,  
    string[] moduleKeys,  
    out TpFault fault);
```

| Needed Permission | | --- |
|-----------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. AuthenticationFailure.TimePunchOutdated AuthenticationFailure.OldDatabaseVersion AuthenticationFailure.NoDatabaseConnection AuthenticationFailure.UnexpectedException |
| authentication | | User authentication |
| moduleKeys | | List of module identifier to check the database structure |

The Fault object returned is of type [TpAuthenticationFault](#)

The following module keys are currently known:

| Schlüssel | Modul |
|-------------|---|
| Datafox | Datafox - Time recording devices |
| Attendance | TimePunch Attendance - Attendance Sheet |
| Cutter | TimePunch Cutter - Working Hour Window |
| Calendar | TimePunch Calendar - Leave Requests |
| SelfService | TimePunch SelfService - Personal data |
| Online | TimePunch Online - Mobile time recording |
| Studio | TimePunch Studio - Time recording at the central service PC |
| Watcher | TimePunch Watcher - Time recording at the own PC |
| Management | TimePunch Management - Manage employees and working time |

TIMEPUNCH API SPECIFICATION 3.4

ValidateAuthentication

This method validates the authentication data and returns the user permissions after the successful validation of the user. Additionally the user license gets validated against the given application keys. If the application keys are null, it gets validated if the user owns at least one application license, independent of which.

```
List<string> ValidateAuthentication (
    out TpFault fault,
    TpAuthentication authentication
    string[] applicationKeys);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Keys of the application for which the license informations shall be validated. |
| return value | | Collection of the user permissions |

The application keys are defined in the class `TimePunch.Enums.Core.ApplicationKeys` as constant values.

ValidateAuthenticationWithProductLine

This method checks the specified credentials and returns user privileges after successful validation. If no application key is specified, the system only checks whether the user has a license at all - regardless of which one it is.

In addition, this method returns the licensed product line.

```
List<string> ValidateAuthenticationWithProductLine(
    out TpFault fault,
    TpAuthentication authentication,
    string[] applicationKeys,
    out ProductLine licensedProductLine);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Keys of the application for which the license informations shall be validated. |

TIMEPUNCH API SPECIFICATION 3.4

| | | |
|----------------------------|------------------------------------|--|
| LicensedProductLine | Out | Productline that is used by the user. Undefined → Currently no product line is set TimePunchPro → Licensed for TimePunch PRO TimePunchOne → Licensed for TimePunch ONE TimePunchTen → Licensed for TimePunch TEN |
| return value | Collection of the user permissions | |

The application keys are defined as constants in the class

`TimePunch.Enums.Core.ApplicationKeys`.

GetCoveredUserProfiles

This method identifies all TimePunch profiles which the user can administrate. All TimePunch profiles that are returned with this method can be used as the identity Parameter for the authentication object.

```
List<UserProfileDto> GetCoveredUserProfiles(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | core@logon |
|-----------------------|---|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| return value | Collection of the user profiles that the given authentication can administrate. | |

GetEntitledUserProfiles

This method returns all TimePunch profiles. This result is independent of the user authentication.

```
List<UserProfileDto> GetEntitledUserProfiles(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | --- |
|-----------------------|---|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| return value | Collection of the user profiles that are authorized to use TimePunch. | |

TIMEPUNCH API SPECIFICATION 3.4

GetLicensedUserProfiles

This method returns all TimePunch profiles, which owns an active and valid license. Additionally the user license gets validated against the given application keys. If the application keys are null, it gets validated if the user owns at least one application license, independent of which.

```
List<UserProfileDto> GetLicensedUserProfiles (
    out TpFault fault,
    TpAuthentication authentication,
    string[] applicationKeys);
```

| Needed Permission | | --- |
|------------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Keys of the application for which the license informations shall be validated. |
| return value | | Collection of the user profiles that own a valid TimePunch license. |

The application keys are defined in the class `TimePunch.Enums.Core.ApplicationKeys` as constant values.

SearchAuthorizedUsers

This method can be used to search for staff members which can logon to the system. That are most users that have access to TimePunch.

```
List<UserProfileDto> SearchAuthorizedUsers(
    out TpFault fault,
    TpAuthentication authentication,
    UserSearchDto userSearchDto);
```

| Needed Permission | | --- |
|-----------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSearchDto | | Search object that specifies what to search for. |
| return value | | Collection of the user profiles that are authorized to use TimePunch. |

TIMEPUNCH API SPECIFICATION 3.4

SearchUserLogons

This method can return extended information to the user profiles. Additionally to the basic data, the permissions and the first / last entry date of the user profile will be returned.

```
List<UserLogonDto> SearchUserLogons(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto userSearchDto)
```

| Needed Permission | | userProfiles@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSearchDto | | Search object that specifies what to search for. |
| return value | | Collection of the user profiles that are authorized to use TimePunch. |

SetPassword

This method is used to set the logon password for a member. The password will only be set for the member that is authenticated through the authentication object.

```
void SetPassword(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string newHashedPwd);
```

| Needed Permission | | password@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| newHashedPwd | | New Password. The password must already been hashed with the MD5 algorithm. |

TIMEPUNCH API SPECIFICATION 3.4

GetAutoRegistration

This method returns internal information about the Auto-Registration feature of TimePunch. This feature allows new employees to log in directly to TimePunch without having an administrator create the profile in TimePunch.

```
AutoregistrationDto GetAutoRegistration(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | --- |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| return value | | Information about the auto-registration feature of TimePunch. |

SaveAutoRegistration

Use this method to set or change the Auto-Registration feature of TimePunch.

```
void SaveAutoRegistration(  
    out TpFault fault,  
    TpAuthentication authentication,  
    AutoregistrationDto autoregistration);
```

| Needed Permission | | core@administrate |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| autoregistration | | Information about the auto-registration feature of TimePunch |

TimePunch Configuration Service

TimePunch's configuration service provides methods for reading and/or saving settings in TimePunch.

LoadCostCenterSettings

This method returns the cost centers defined in TimePunch for the different posting types.

```
CostCenterDto LoadCostCenterSettings(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | core@logon |
|-----------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | Information about the cost centers |

SaveCostCenterSettings

This method allows you to save the cost centers for the different booking types in TimePunch.

```
void SaveCostCenterSettings(  
    out TpFault fault,  
    TpAuthentication authentication,  
    CostCenterDto costCenter);
```

| Needed Permission | | datev@export |
|-----------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| costCenter | | Information about the cost centers |

TIMEPUNCH API SPECIFICATION 3.4

LoadDatevExportSettings

This method loads the settings for exporting to DATEV.

```
DatevExportDto LoadDatevExportSettings(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | Information about the datev settings |

SaveDatevExportSettings

This method saves the settings for the DATEV export.

```
void SaveDatevExportSettings(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DatevExportSaveDto datevExport);
```

| Needed Permission | | datev@export |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| datevExport | | Information about the datev settings |

LoadLexwareExportSettings

This method loads the settings for exporting to LEXWARE..

```
LexwareExportDto LoadLexwareExportSettings(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | Information about the Lexware settings |

TIMEPUNCH API SPECIFICATION 3.4

SaveLexwareExportSettings

This method saves the settings for the LEXWARE export.

```
void SaveLexwareExportSettings(  
    out TpFault fault,  
    TpAuthentication authentication,  
    LexwareExportDto LexwareExport);
```

| Needed Permission | | lexware@export |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| LexwareExport | | Information about the Lexware settings |

LoadUserSettings

This method loads settings that affect all users equally and cause a special behavior in TimePunch.

```
UserSettingsDto LoadUserSettings(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | User settings object |

TIMEPUNCH API SPECIFICATION 3.4

SaveUserSettings

This method saves the settings for the special behavior in TimePunch.

```
void SaveUserSettings(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSettingsDto userSettings);
```

| Needed Permission | | core@administrate |
|-----------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSettings | | Information about the datev settings |

TimePunch Common Task Service

This service contains methods in order to work with common tasks of TimePunch. Common Tasks can be used in any project.

GetActiveCommonTasks

This method loads all common tasks that are active. Active tasks are such which have no delete flag set.

```
List<TaskDto> GetActiveCommonTasks (  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | commonTasks@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | Collection of active common tasks |

GetAllCommonTasks

This method load all common tasks, independently if they are marked as deleted or not.

```
List<TaskDto> GetAllCommonTasks (  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | commonTasks@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | Collection of common tasks |

TIMEPUNCH API SPECIFICATION 3.4

ImportTasks

This method imports the transferred activities. In contrast to the Save method, the activities are not mapped by ID, but by name.

```
List<TaskDto> ImportTasks(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<TaskDto> tasks);
```

| Needed Permission | | commonTasks@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| tasks | | Collection of common tasks to import |
| Return Value | | Returns a collection of all common tasks |

LoadCommonTask

This method loads a single task. This can be useful if one has to reload a deleted task for a single time entry.

```
TaskDto LoadCommonTask(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid taskId);
```

| Needed Permission | | commonTasks@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | The loaded task |

TIMEPUNCH API SPECIFICATION 3.4

SaveCommonTasks

This method stores the common task list to the database. It's important that the list contains all active tasks. Tasks that are not within the list, will be marked as deleted automatically.

```
void SaveCommonTasks(  
    out TpFault fault,  
    TpAuthentication authentication,  
    IEnumerable<TaskDto> tasks);
```

| Needed Permission | | commonTasks@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| tasks | | Collection of active common tasks |

TimePunch Customer Service

This service provides an interface for accessing the customer data that are stored in TimePunch.

CreateCustomer

This method creates a new customer object and returns it. The customer data won't be stored in the database.

```
CustomerDto CreateCustomer(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | customers@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | New CustomerDTO object with the preset of customer data. |

SaveCustomers

This method saves the customer to the database and returns the stored data as a result again.

```
CustomerDto[] SaveCustomers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    CustomerDto[] customers);
```

| Needed Permission | | customers@import |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| customers | | Array of customer data to store in the database |
| Return value | | The customer data that has been stored in the database. |

TIMEPUNCH API SPECIFICATION 3.4

ImportCustomers

This method imports the given customer data in the database. The difference to the save method is that the customer data is identified with the customers Ref Nr and the Name, instead of the Id.

```
CustomerDto[] ImportCustomers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    CustomerDto[] customers);
```

| Needed Permission | | customers@import |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| customers | | Array of customer data to import in the database |
| Return value | | The customer data that has been stored in the database. |

LoadCustomers

This method loads the customer data with the passed customer ids. The method should be preferred to the search, if the IDs of the customer data are known.

```
CustomerDto[] LoadCustomers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] customerIds);
```

| Needed Permission | | customers@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| customerIds | | Array of customer ids to load from the database |
| Return value | | The customer data that has been loaded from the database. |

TIMEPUNCH API SPECIFICATION 3.4

MarkCustomersAsDeleted

This method marks the customer data as deleted. But the data won't be physically deleted.

```
void MarkCustomersAsDeleted(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] customerIds);
```

| Needed Permission | | customers@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| customerIds | | Array of customer ids to mark as deleted in database |

DeleteCustomers

This method attempts to physically delete the customer's data in the database. If the customer is already in use, the physical deletion is no longer possible. In this case, the customer should only be marked as deleted.

```
void DeleteCustomers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] customerIds);
```

| Needed Permission | | customers@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| customerIds | | Array of customer ids to physically delete in database |

TIMEPUNCH API SPECIFICATION 3.4

SearchCustomers

This method searches for all customers on the basis of the specified search object.

```
CustomerDto[] SearchCustomers(  
    out Tpfault fault,  
    TpAuthentication authentication,  
    CustomerSearchDto templateCustomer);
```

| Needed Permission | | customers@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| templateCustomer | | Search object that holds the properties used for searching the real customers |
| Return value | | Found customer data |

SearchActiveCustomers

This method searches for all active customers on the basis of the specified search object.

```
CustomerDto[] SearchCustomers(  
    out Tpfault fault,  
    TpAuthentication authentication,  
    CustomerSearchDto templateCustomer);
```

| Needed Permission | | customers@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| templateCustomer | | Search object that holds the properties used for searching the real customers |
| Return value | | Found customer data |

TimePunch Holiday Service

This service gains access to the public holiday definitions in TimePunch.

GetCountries

Using this method all countries and regions, that are stored in TimePunch for holiday calculation, will be returned.

```
CountryDto[] GetCountries(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | publicHolidays@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return Value | | List of all defined Countries |

GetPublicHolidays

This method loads all holidays of the given country with the defined year.

```
PublicHolidayDto[] GetPublicHolidays(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid countryId,  
    int year);
```

| Needed Permission | | publicHolidays@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| countryId | | Id of the country to load the holidays |
| year | | Year to load the holidays |
| Return Value | | List of all defined holidays |

TIMEPUNCH API SPECIFICATION 3.4

SavePublicHolidays

This method saves the holidays passed to the country and year defined.

```
void SavePublicHolidays(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid countryId,  
    int year,  
    PublicHolidayDto[] publicHolidayDtos)
```

| Needed Permission | | publicHolidays@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| countryId | | Id of the country to load the holidays |
| year | | Year to save the holidays |

ExportPublicHolidays

This method exports all public holidays from TimePunch. The difference to the load method is that not only the holidays of the loaded year and country, but all holidays are returned.

```
PublicHolidayDto[] ExportPublicHolidays(  
    out TpFault fault,  
    TpAuthentication authentication)
```

| Needed Permission | | publicHolidays@export |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return Value | | List of all defined holidays |

TIMEPUNCH API SPECIFICATION 3.4

ImportPublicHolidays

This method imports all public holidays from TimePunch. The difference to the storage method is that not only the holidays of the loaded year and country, but all holidays are stored.

```
void ImportPublicHolidays(  
    out TpFault fault,  
    TpAuthentication authentication,  
    PublicHolidayDto[] publicHolidayDtos)
```

| Needed Permission | | publicHolidays@import |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| publicHolidayDtos | | List of public holidays to import |

TimePunch Licensing Service

This service can be used to add license information for a TimePunch Profile.

AddLicense

This method adds a TimePunch license to the internal license management.

```
ApplicationLicenseDto AddLicense(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string applicationKey,  
    string licensedFor,  
    string licenseCode);
```

| Needed Permission | | core@logon |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKey | | Currently not used! |
| licensedFor | | License Name |
| licenseCode | | License Code |
| return value | | Information about the currently added license |

GetApplicationLicenses

This method returns the application licenses for the given application keys.

```
ApplicationLicenseDto[] GetApplicationLicenses(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to retrieve license information for. |
| return value | | Application specific license information |

TIMEPUNCH API SPECIFICATION 3.4

GetLicensingState

This method checks the licensing of a single employee. It is also important to specify which licenses you want to search for. This is determined by the ApplicationKeys parameter. The actual licensed application key is returned. Example, a license for TimePunch PRO and PZE Watcher is searched for. If the employee has licensed the PZE Watcher, the application key for the Watcher is returned.

```
LicensingState GetLicensingState(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    out int daysLeft,  
    out string applicationKey);
```

| Needed Permission | | core@logon |
|------------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to retrieve license information for. |
| daysLeft | Out | Defines how many days the license will be valid. |
| applicationKey | Out | Gets the application key for that the user owns a specific license. |
| return value | | Gets the licensing state of the user NoLicense, TestLicenseFound, TestLicenseExpired, LicenseFound, LicenseExpired, Undefined |

TIMEPUNCH API SPECIFICATION 3.4

IsLicenseUpgradeable

This method checks whether the existing licenses can be updated to a new version of TimePunch. This is of particular interest for version-specific licenses, since these licenses are only valid for a certain version.

```
bool IsLicenseUpgradeable(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    string version);
```

| Needed Permission | | --- |
|------------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to retrieve license information for. |
| version | | Version that shall be tested. |
| return value | | True, if the current license is upgradable. E.g. the current version is 2.95 - the new version will be 3.2 which will then result in true as the return value. |

RequestUpgradeOffer

If the result of the upgrade check is true (IsLicenseUpgradeable), you can use this method to request a quotation for the version upgrade. The method connects to TimePunch and transfers the license data so that an individual upgrade offer can be sent.

```
RequestOfferResult RequestUpgradeOffer(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    RequestUpgradeOfferDto requestUpgradeOffer);
```

| Needed Permission | | --- |
|----------------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to retrieve license information for. |
| requestUpgradeOffer | | Customer data used to send the upgrade offer |
| Return value | | Result of the upgrade offer request Error, NoUpgradeNecessary, AutomaticOffer, ManualOffer |

TIMEPUNCH API SPECIFICATION 3.4

GetUserLicenses

This method returns all license data for the employees specified in the search object. It is important to specify which licenses to search. This is set by the parameter applicationKeys.

```
ApplicationKeys.UserLicenseDto[] GetUserLicenses(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    UserSearchDto searchDto);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to retrieve license information for. |
| searchDto | | Search object to specify the users for loading the license data. |
| return value | | User specific license information |

RemoveLicenses

Using this method already inserted application licenses can be removed again.

```
void RemoveLicenses(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    Guid[] applicationLicenseIds);
```

| Needed Permission | | licenses@manage |
|-----------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to register the user |
| applicationLicenseIds | | List containing the application license ids that shall be removed for the given application keys. |

TIMEPUNCH API SPECIFICATION 3.4

RegisterUsers

This method attempts to register the specified employee profiles for the application specified in the parameter applicationKeys. This will only work if for the employees still licenses are available.

```
void RegisterUsers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    List<Guid> userIds);
```

| Needed Permission | | core@logon |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to register the user |
| userIds | | List containing the user ids that shall be registered for the given application keys. |

UnregisterUsers

This method removes the license information for an employee. This removes only the licenses for the applications that are specified in the parameter applicationKeys.

```
void UnregisterUsers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string[] applicationKeys,  
    List<Guid> userIds);
```

| Needed Permission | | core@logon |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| applicationKeys | | Specifies the application to un-register the user |
| userIds | | List containing the user ids that shall be un-registered for the given application keys. |

TimePunch Logging Service

This service is used to start and stop the time recording and to administrate the recording settings.

GetActiveTimeEntry

This method returns the current active time entry. If no time recording is active, it returns NULL.

```
ActiveTimeEntryDto GetActiveTimeEntry(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | logging@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| return value | | The active time entry or NULL, if the time tracking has not been started. |

SearchActiveTimeEntries

This method searches for active time-entries with the given filter. The permission of the authenticated user will be considered. Only time entries of the user profiles that the user is allowed to access will be returned.

```
List<ActiveTimeEntryDto> SearchActiveTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ActiveTimeEntrySearchDto search);
```

| Needed Permission | | timeEntries@access / core@switchIdentity |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| search | | The filter that is used to search the time entries |
| Return value | | Collection with the found time entries |

TIMEPUNCH API SPECIFICATION 3.4

StartWorkTimeLogging

This method starts the work time recording.

```
ActiveTimeEntryDto StartWorkTimeLogging(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid projectId,  
    Guid? taskId,  
    Guid? customerId,  
    string description,  
    DateTime? startTime,  
    out List<TimeEntryDto> todayEntries);
```

| Needed Permission | | startTime == null ? logging@access : logging@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectId | | The id of the project that is used to start the time recording |
| taskId | | The id of the task that is used for the time recording |
| customerId | | The id of the customer that is used for the time recording |
| description | | The description that will be used for the time recording |
| startTime | | The start time of the time recording. Regularly this should be NULL; otherwise a higher user-permission is needed. |
| todayEntries | Out | Collection of all entries of the current day. |
| Return value | | The current time entry that has been started by the method. |

TIMEPUNCH API SPECIFICATION 3.4

StartBreakTimeLogging

This method starts a new break time recording. It's important that this can only be done if an active time recording is already in place. A break can only be started as a part of the current work time recording.

```
ActiveTimeEntryDto StartBreakTimeLogging(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | logging@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| return value | | The current time entry that has been modified by the method. |

EndLogging

This method ends the current recording and commits the recorded time entry for the user.

```
ActiveTimeEntryDto EndLogging(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? endTime,  
    out TimeSpan? committedWorkTime,  
    out List<TimeEntryDto> todayEntries);
```

| Needed Permission | | endTime == null ? logging@access : logging@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| endTime | | The end time of the time recording. Regularly this should be NULL; otherwise a higher user-permission is needed. |
| committedWorkTime | Out | The committed work time |
| todayEntries | Out | Collection of all entries of the current day. |
| Return value | | The current time entry that has been committed by the method. |

TIMEPUNCH API SPECIFICATION 3.4

CancelLogging

This method cancels the current work time recording. This can be necessary if the current work entry has not been ended correctly.

```
void CancelLogging(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | logging@access |
|-----------------------|----------|--|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |

LockSession

This method locks the current work time recording. This should be done e.g. when the screensaver is activated. This locking does not mean an instant break of the current work time recording. How the lock is handled will be defined with the locking settings.

```
void LockSession(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? lockTime);
```

| Needed Permission | | logging@access |
|-----------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| lockTime | | Time that is used as the current lock time. If this parameter is null, the time at the server will be taken. |

TIMEPUNCH API SPECIFICATION 3.4

UnlockSession

This method unlocks the previously locked time entry. When unlocking the time entry, TimePunch analyses the lock time and decides how to handle the current work time recording.

```
ActiveTimeEntryDto UnlockSession(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? unlockTime);
```

| Needed Permission | | logging@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| unlockTime | | Time that is used as the current unlock time. If this parameter is null, the time at the server will be taken. |
| Return Value | | The unlocked time entry or NULL, if no time entry is active after unlocking the session. |

SaveLoggingSettings

This method stores the recording settings for the TimePunch profile.

```
void SaveLoggingSettings(  
    out TpFault fault,  
    TpAuthentication authentication,  
    LoggingSettingsDto settingsDto);
```

| Needed Permission | | logging@settings |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| settingsDto | | The amended Logging Settings that shall be stored. |

SearchLoggingSettings

This method returns the logging settings for the given search object.

```
List<LoggingSettingsDto> SearchLoggingSettings(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto userSearchDto);
```

| Needed Permission | | logging@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSearchDto | | Search object containing the user search data. The coverage will be applied accordingly to the user authentication. |
| Return value | | All user logging settings. |

GetLoggingContext

This method is the central place to load all data that are necessary in order to get an overview of the complete time recording status.

```
LoggingContextDto GetLoggingContext(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | logging@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | All data regarding the current time entry logging. |

TIMEPUNCH API SPECIFICATION 3.4

ValidateAndSaveActiveTimeEntry

This method is used to change the currently active time entry. It can be used to change the project, switch start and end time or to amend the project.

```
ActiveTimeEntryDto ValidateAndSaveActiveTimeEntry(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ActiveTimeEntryDto activeTimeEntry);
```

| Needed Permission | | logging@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| activeTimeEntry | | The active time entry that shall be stored |
| Return value | | The currently active time entry after the amendments |

ValidateAndSaveRestrictedActiveTimeEntry

This method is used to amend the currently active time entry. Because the method needs less permission one can only change the description and the project or task.

```
ActiveTimeEntryDto ValidateAndSaveRestrictedActiveTimeEntry(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntryRestrictedSaveDto activeTimeEntry);
```

| Needed Permission | | logging@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| activeTimeEntry | | The active time entry that shall be stored |
| Return value | | The currently active time entry after the amendments |

TimePunch Mailing Service

This service contains methods in order to send mails within TimePunch.

SendMail

This method sends an email within TimePunch. The sender of the email is always the authorized employee.

```
void SendMail(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string subject,  
    string body,  
    IEnumerable<UserMailAddressDto> userMailAddresses,  
    IEnumerable<PlainMailAddressDto> plainMailAddresses);
```

| Needed Permission | | mails@access |
|--------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| subject | | Mail subject |
| body | | Mail body |
| userMailAddresses | | List containing all TimePunch users that are Receiver or Copy Receiver of the mail. |
| plainMailAddresses | | List containing all non - TimePunch users that are Receiver or Copy Receiver of the mail. |

TIMEPUNCH API SPECIFICATION 3.4

SendMailWithAttachments

This method sends a mail within TimePunch with appropriate attachments. The sender of the email is always the authorized employee.

```
void SendMailWithAttachments(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string subject,  
    string body,  
    IEnumerable<UserMailAddressDto> userMailAddresses,  
    IEnumerable<PlainMailAddressDto> plainMailAddresses,  
    IEnumerable<MailAttachementDto> mailAttachments);
```

| Needed Permission | | mails@access |
|--------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| subject | | Mail subject |
| body | | Mail body |
| userMailAddresses | | List containing all TimePunch users that are Receiver or Copy Receiver of the mail. |
| plainMailAddresses | | List containing all non - TimePunch users that are Receiver or Copy Receiver of the mail. |
| mailAttachments | | List containing all attachments for the mail. |

TimePunch Project Service

This service contains methods in order to work with the projects of TimePunch.

CreateProject

This method creates a new project and returns it. The project is saved directly in the database when it is created and filled with the default values.

```
ProjectDto CreateProject(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string projectId,  
    string projectDescription,  
    string projectCode,  
    Guid? copyFromProject);
```

| Needed Permission | | projects@manage |
|--------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectId | | Unique identifier of the project |
| projectDescription | | Short description of the project |
| projectCode | | Unique identifier that can be used by a barcode reader |
| Guid? | | Id of the project that shall be used as a template, or NULL. |
| Return value | | Returns the created project |

DeleteProject

This method deletes the given project physically in the database. But this can only be done, if the project is not referenced by any time-entries. If the project is referenced it's recommended to do a soft-delete. That means to set the delete flag in the project entity.

```
void DeleteProject(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid projectId);
```

| Needed Permission | | projects@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectId | | Id of the project that shall be deleted |

TIMEPUNCH API SPECIFICATION 3.4

ExportProjectUserRelations

This method exports the employee-specific settings for one or more projects.

```
List<ProjectUserRelationDto> ExportProjectUserRelations(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ProjectSearchDto projectSearch);
```

| Needed Permission | | projects@export |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| searchCriteria | | Criteria that are used for searching the projects. |
| Return value | | Collection of project related user data |

GetProjectUsers

This method returns a list of TimePunch profiles that are able to book on certain projects

```
List<ProjectUserDto> GetProjectUsers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] projectIds);
```

| Needed Permission | | projects@report |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectIds | | Collection of project ids to load the project relevant times |
| Return value | | List of project / user combinations that are allowed to book the relevant project ids. It also contains the scheduled work and the pool type (Shared / Private) |

TIMEPUNCH API SPECIFICATION 3.4

GetTotalProjectTimes

This method is used to retrieve the current project times for the given projects and employees.

```
List<ProjectTimeDto> GetTotalProjectTimes(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] projectIds,  
    Guid[] userIds);
```

| Needed Permission | | UserIds == null ? projects@access : projects@report |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectIds | | Collection of project ids to load the project relevant times |
| userIds | | User ids of all users for which the project data shall be loaded, or null to load the data for the current user. |

ImportProjects

This method imports a list of projects. In contrast to the Save method, it is not the project IDs that are used for identification, but the project name.

```
void ImportProjects(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<ProjectDto> projects);
```

| Needed Permission | | projects@import |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projects | | Collection of projects to import |

ImportProjectTasks

This method imports a list of project-specific activities. The activities are identified by the description text and not by the task ID.

```
void ImportProjectTasks(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<ProjectTaskDto> projectTasks);
```

| Needed Permission | | projects@import |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectTasks | | Collection of project dependend tasks to import |

ImportProjectUserRelations

This method imports the employee assignments to a project. In contrast to the *SaveProjectUserRelations* method, the employees are not assigned to the logon name via the technical ID, but via the logon name.

```
void ImportProjectUserRelations(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<ProjectUserRelationDto> userRelations);
```

| Needed Permission | | projects@import |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userRelations | | Collection of project user relations tasks to import |

TIMEPUNCH API SPECIFICATION 3.4

LoadProject

This method loads the project with the given project id.

```
ProjectDto LoadProject(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid projectId);
```

| Needed Permission | | projects@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectId | | Id of the project to load |
| Return value | | The loaded project data |

LoadProjectUserRelations

This method loads the employee-specific settings of a project for all authorized employees.

```
List<ProjectUserRelationDto> LoadProjectUserRelations(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid projectId);
```

| Needed Permission | | projects@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectId | | Id of the project to load |
| Return value | | Collection of user specific project data |

MarkProjectsAsDone

This method sets the delete-flag in the project and marks the project therefore as finished. That means it's not allowed to book new time-entries to the project. But existing time-entries stay as they are.

```
void MarkProjectsAsDone(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] projectIds);
```

| Needed Permission | | projects@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectIds | | Collection of project ids to set the soft delete flag |

SaveProject

This method stores the project in the database.

```
ProjectDto SaveProject(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ProjectDto project);
```

| Needed Permission | | projects@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| project | | Project data that shall be stored |
| Return value | | The stored project data will be returned |

SaveProjectUserRelations

This method updates the employee assignment in the projects.

```
void SaveProjectUserRelations(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<ProjectUserRelationSaveDto> userRelations);
```

| Needed Permission | | projects@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userRelations | | List of user and project related data. |

SearchAuthorizedProjects

This method searches for projects in which the employee is authorized, using the specified search criteria. The found projects are returned. Authorized projects are all projects assigned to the employee.

```
List<ProjectDto> SearchAuthorizedProjects(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ProjectSearchDto searchCriteria);
```

| Needed Permission | | projects@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| searchCriteria | | Criteria that are used for searching the projects. |
| Return value | | Collection of found projects |

SearchProjects

This method searches all available projects with the given search criteria. The found projects will be returned.

```
List<ProjectDto> SearchProjects(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ProjectSearchDto searchCriteria);
```

| Needed Permission | | projects@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| searchCriteria | | Criteria that are used for searching the projects. |
| Return value | | Collection of found projects |

TIMEPUNCH API SPECIFICATION 3.4

SetProjectLifeCycle

This method sets the current lifecycle of one or more projects. The lifecycle specifies how the project can be used in TimePunch.

```
public enum ProjectLifeCycle
{
    /// Life cycle is undefined
    Undefined,

    /// The project can be changed, but not booked
    Planning,

    /// The project can be changed and booked
    Active,

    /// The project can not be changed, not booked, but reported
    Done,

    /// The project can not be changed, not booked and is not visible for reporting
    Deleted,
}

void SetProjectLifeCycle(
    out TpFault fault,
    TpAuthentication authentication,
    Guid[] projectIds,
    ProjectLifeCycle projectLifeCycle);
```

| Needed Permission | | projects@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projectIds | | Collection of project ids for that the life cycle shall be changed. |
| projectLifeCycle | | The new life cycle to set |

TimePunch Reporting Service

The reporting service provides methods for reporting in TimePunch. The methods are very powerful and should be used only for reporting purposes due to the performance.

GetDailySummaries

This method returns a daily summary of working time data for the selected period and given employees. The coverage applies to the used identity data.

```
List<ReportDailySummaryDto> GetDailySummaries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime startDate,  
    DateTime endDate,  
    List<Guid> userIds);
```

| Needed Permission | | summary@report |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| startDate | | First reporting date |
| endDate | | Last reporting date |
| userIds | | List containing all user ids for which the daily summary shall be reported. |
| Return value | | List of all daily summaries. |

GetMonthlySummaries

This method returns a monthly summary of the working time data for the selected period and the transferred employees.

```
List<ReportSummaryDto> GetMonthlySummaries(  
    out TpFault fault,  
    TpAuthentication authentication, >  
    DateTime startDate,  
    DateTime endDate,  
    List<Guid> userIds);
```

| Needed Permission | | summary@report |
|-------------------|----------|---|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| startDate | | First reporting date |
| endDate | | Last reporting date |
| userIds | | List containing all user ids for which the daily summary shall be reported. |

TIMEPUNCH API SPECIFICATION 3.4

| | | |
|-------------------|----------|--------------------------------|
| Needed Permission | | summary@report |
| Name | Modifier | Description |
| Return value | | List of all monthly summaries. |

GetYearlySummaries

This method returns an annual summary of the working time data for the selected period and the transferred employees.

```
List<ReportYearlySummaryDto> GetYearlySummaries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime startDate,  
    DateTime endDate,  
    List<Guid> userIds);
```

| | | |
|-------------------|----------|---|
| Needed Permission | | summary@report |
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| startDate | | First reporting date |
| endDate | | Last reporting date |
| userIds | | List containing all user ids for which the daily summary shall be reported. |
| Return value | | List of all yearly summaries. |

TIMEPUNCH API SPECIFICATION 3.4

GetDailySummariesAnonymized

The difference here is that the working time data is largely anonymized, so that it cannot be deduced from the actual times of the employees. This means that the method also requires less authorization.

```
List<ReportDailySummaryDto> GetDailySummariesAnonymized(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime startDate,  
    DateTime endDate,  
    List<Guid> userIds);
```

| Needed Permission | | summary@reportAnon |
|-----------------------|----------|---|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| startDate | | First reporting date |
| endDate | | Last reporting date |
| userIds | | List containing all user ids for which the daily summary shall be reported. |
| Return value | | List of all daily summaries. |

GetReportTimeEntriesUserBased

This method returns the time entries of the selected employees. The focus of the query is on employee times, not on one or more projects.

```
List<ReportTimeEntryDto> GetReportTimeEntriesUserBased(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySearchDto searchDto);
```

| Needed Permission | | timeEntries@report |
|-----------------------|----------|--|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| searchDto | | Search object to filter the time entries |
| Return value | | List of all selected time entries |

TIMEPUNCH API SPECIFICATION 3.4

GetReportTimeEntriesProjectBased

This method returns the time entries of the selected projects. In contrast to the *GetReportTimeEntriesUserBased* method, the focus is not on employees, but on projects. The time accounts of the projects (total, posted, available) are determined for each project on the transferred key date. Nevertheless, the method also selects the postings assigned to the projects and returns them.

```
List<ReportTimeEntryDto> GetReportTimeEntriesProjectBased(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? reportingDate,  
    out List<ReportProjectTimeDto> projectTimes,  
    TimeEntrySearchDto searchDto);
```

| Needed Permission | | projects@report |
|-------------------|----------|---|
| Name | Modifier | Description |
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| reportingDate | | Reference Date that is used to evaluate the project counters. If null, the current date will be used. |
| projectTimes | Out | List of all project times at the given reporting date. |
| searchDto | | Search object to filter the time entries |
| Return value | | List of all selected time entries |

GetReportProjectTimes

This method returns all project times of the passed project IDs for the given key date.

```
List<ReportProjectTimeDto> GetReportProjectTimes(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? reportingDate,  
    List<Guid> projectIds);
```

| Name | Modifier | Description |
|----------------|----------|---|
| Fault | Out | Contains the error if an exception occurs. |
| Authentication | | User authentication |
| reportingDate | | Reference Date that is used to evaluate the project counters. If null, the current date will be used. |
| projectTimes | Out | List of all project times at the given reporting date. |
| Return value | | List of all project times at the given reporting date. |

TimePunch Summary Service

This service contains methods in order to access the time accounts of the TimePunch profiles.

GetSummaryLocks

This method returns all information about the month-end closings that are found for the employees through the search object.

```
List<SummaryLockDto> GetSummaryLocks(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto userFilter = null);
```

| Needed Permission | | monthend@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userFilter | | Search object containing the user filter. The coverage will be applied to the filter. |
| Return value | | List that contains all summary locks for the found staff members. |

LoadAllYearlySummaries

This method returns all yearly account data for the authorized employee.

```
List<YearlySummaryDto> LoadAllYearlySummaries(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | summary@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | List containing all yearly sumamries for the given user. |

TIMEPUNCH API SPECIFICATION 3.4

LoadDailySummaries

Using this method a summary for each working day in the given period can be loaded.

```
List<DailySummaryDto> LoadDailySummaries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime startDate,  
    DateTime endDate);
```

| Needed Permission | | summary@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| startDate | | First date to retrieve the daily summary |
| endDate | | Last date to retrieve the daily summary |
| Return value | | List that contains all daily summaries from start through end date |

LoadMonthlySummary

This method will load the monthly summary of the working time for the specified user at the given reference date.

```
MonthlySummaryDto LoadMonthlySummary(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? referenceDate = null);
```

| Needed Permission | | summary@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| referenceDate | | Date for that the monthly Summary shall be loaded, or null if the monthly summary for the current date shall be returned. |
| Return value | | The monthly summary for the given reference date. |

TIMEPUNCH API SPECIFICATION 3.4

LoadTimeAccounts

This method loads the current time accounts of the TimePunch profiles. Depending on whether only the time account of the authorized user or all time accounts may accessed, different permissions will be needed.

```
List<TimeAccountDto> LoadTimeAccounts(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto userSearch,  
    DateTime? referenceDate = null);
```

| Needed Permission | | timeAccounts@access / timeAccounts@report |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSearch | | Information that are used to search the TimePunch profiles |
| referenceDate | | The date of which the time accounts shall be loaded. By default this is always the day before today |
| Return value | | List of Time account information entities |

LoadYearlySummary

This method loads the annual vacations for the authorized user for the given reference date.

```
YearlySummaryDto LoadYearlySummary(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? referenceDate = null);
```

| Needed Permission | | summary@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| referenceDate | | The date of which the yearly summary shall be loaded, or null if the current date is used. |
| Return value | | The yearly summary for the given user and reference date. |

TIMEPUNCH API SPECIFICATION 3.4

ReCalculateAllSummaries

This method forces the recalculation of all monthly data of the employees specified in the user filter starting from the reference date.

```
void ReCalculateAllSummaries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto userFilter,  
    DateTime? referenceDate = null)
```

| Needed Permission | | summary@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userFilter | | Filter that is used to search the TimePunch profiles. |
| referenceDate | | Defines the date since when the monthly summaries shall be re-calculated. If NULL, all monthly summaries of a user will be re-calculated. |
| Return value | | The calculated monthly summary, after the calculation date has been applied. |

SaveMonthlySummary

This method saves the basic data that are used to calculate the monthly working and flexi time.

```
MonthlySummaryDto SaveMonthlySummary(  
    out TpFault fault,  
    TpAuthentication authentication,  
    MonthlySummarySaveDto summary);
```

| Needed Permission | | summary@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| summary | | Contains the data that can be used to calculate the monthly working time data. |
| Return value | | The calculated monthly summary, after the calculation date has been applied. |

TIMEPUNCH API SPECIFICATION 3.4

SaveYearlySummary

This method save the basic data that are used to calculate the annual vacation.

```
YearlySummaryDto SaveYearlySummary(  
    out TpFault fault,  
    TpAuthentication authentication,  
    YearlySummarySaveDto summary);
```

| Needed Permission | | summary@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| summary | | Contains the data that can be used to calculate the annual leave. |
| Return value | | The calculated annual leave, after the calculation date has been applied. |

SetSummaryLocks

Using this method, one can perform the month-end closing. The conclusion is either performed for the currently authenticated- employee (if the UserIDs are NULL), or for all given employee ids.

```
void SetSummaryLocks(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? lockedUpTo,  
    List<Guid> userIds = null);
```

| Needed Permission | | CAN_MANAGE_MONTHEND, if lockedUpTo is filled CAN_RESET_MONTHEND, if lockedUpTo is NULL |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| lockedUpTo | | Date up to that the month ends gets locked. If the lockedUpTo is NULL all months will be unlocked. |
| userIds | | A list that contains the user ids for that the month end closing will be executed. If the list is NULL or empty, the current identity user will be used. |

TimePunch Sync Service

This service contains methods in order to search for time entries or projects that have to be synchronized with mobile devices. Also it offers method in order to import / synchronize time entries and projects.

GetUseableSyncUserProfiles

This method searches for all TimePunch profiles that can be used in order to sync data. As an additional information the last synchronization date will be returned.

```
List<SyncUserProfileDto> GetUseableSyncUserProfiles(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | sync@access - current user core@switchIdentity - different users |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | List of TimePunch profiles that can be used for synchronisation purposes. |

TIMEPUNCH API SPECIFICATION 3.4

SearchTimeEntriesForSynchronisation

This method searches all time entries and the depending projects as well as the tasks that can be send to a mobile device since the given synchronisation time.

The method does not check the creation or modification date, but the date of the time entry if it's newer than the sync time.

```
void SearchTimeEntriesForSynchronisation(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime? lastSyncTime,  
    out IEnumerable<TimeEntryDto> timeEntries,  
    out IEnumerable<ProjectDto> projects,  
    out IEnumerable< TaskDto> commonTasks);
```

| Needed Permission | | timeEntries@upload |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| lastSyncTime | | Synchronisation-Time |
| timeEntries | Out | Collection of time entries starting with the given sync time |
| projects | Out | Projects that are referenced by the found time entries |
| commonTasks | Out | Common tasks that are referenced by the found time entries |

SearchProjectsForSynchronisation

This method searches for all projects that a user can send to his mobile devices.

```
IEnumerable<ProjectDto> SearchProjectsForSynchronisation(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | projects@upload |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return Value | | Projects that can be used for synchronization purposes. |

TIMEPUNCH API SPECIFICATION 3.4

UpdateLastSyncDate

This method updates the synchronisation date for the current user.

```
void UpdateLastSyncDate(  
    out TpFault fault,  
    DateTime? syncDate,  
    TpAuthentication authentication);
```

| Needed Permission | | timeEntries@upload |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| syncDate | | Sync Date to set, or null if the current date shall be used. |
| authentication | | User authentication |

ApplyProjects

This method imports the given projects and merges it with the already existing.

```
void ApplyProjects(  
    out TpFault fault,  
    TpAuthentication authentication,  
    IEnumerable<ProjectDto> projects,  
    IEnumerable<TaskDto> commonTasks);
```

| Needed Permission | | projects@download |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| projects | | Collection of projects that shall be imported. |
| commonTasks | | Collection of common tasks that shall be imported. |

TIMEPUNCH API SPECIFICATION 3.4

ApplyTimeEntries

This method imports the given time-entries, projects as well as tasks and merges it with the already existing.

```
void ApplyTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    IEnumerable<TimeEntrySaveDto> timeEntries,  
    IEnumerable<ProjectDto> projects,  
    IEnumerable<TaskDto> commonTasks);
```

| Needed Permission | | projects@download |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntries | | Collection of time entries to impoert |
| projects | | Collection of projects that are referenced by the time entries. |
| commonTasks | | Collection of common tasks that are referenced by the time entries. |

TimePunch Async Sync Service

This service offers methods for the asynchronous import of time entries or projects. The asynchronous import has the advantage that the client can be informed about the current state of the import via the offered call back service. This can be very useful with big and long running imports.

ApplyProjects

This method imports the given projects and merges it with the already existing.

```
void ApplyProjects(  
    TpAuthentication authentication,  
    IEnumerable<ProjectDto> projects,  
    IEnumerable<TaskDto> commonTasks);
```

| Needed Permission | | projects@download |
|-------------------|----------|--|
| Name | Modifier | Description |
| authentication | | User authentication |
| projects | | Collection of projects that shall be imported. |
| commonTasks | | Collection of common tasks that shall be imported. |

ApplyTimeEntries

This method imports the given time-entries, projects as well as tasks and merges it with the already existing.

```
void ApplyTimeEntries(  
    TpAuthentication authentication,  
    IEnumerable<TimeEntrySaveDto> timeEntries,  
    IEnumerable<ProjectDto> projects,  
    IEnumerable<TaskDto> commonTasks);
```

| Needed Permission | | projects@download |
|-------------------|----------|---|
| Name | Modifier | Description |
| authentication | | User authentication |
| timeEntries | | Collection of time entries to impoert |
| projects | | Collection of projects that are referenced by the time entries. |
| commonTasks | | Collection of common tasks that are referenced by the time entries. |

TIMEPUNCH API SPECIFICATION 3.4

ITpSyncCallbackService

In order to use the call back service, the ITpSyncCallbackService Interface of the client needs to be implemented.

Code Snippet

```
[ServiceContract]
public interface ITpSyncCallbackService
{
    /// <summary>
    /// Informs the Client that the current sync failed with an exception
    /// </summary>
    /// <param name="fault">The fault.</param>
    [OperationContract (IsOneWay=true)]
    void SyncFailed(TpFault fault);

    /// <summary>
    /// Informs the Client that the current sync succeeded properly
    /// </summary>
    [OperationContract (IsOneWay = true)]
    void SyncSucceeded();

    /// <summary>
    /// Informs the Client about the progress of the sync process
    /// </summary>
    [OperationContract (IsOneWay = true)]
    void SyncProgress(int percent);
}
```

TimePunch TimeEntry Service

This service contains methods in order to work with the time-entries of TimePunch.

AssignTimeEntriesToCustomer

With this method, you can assign multiple time entries to a new customer.

```
void AssignTimeEntriesToCustomer(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] timeEntryIds,  
    Guid customerId);
```

| Needed Permission | | timeEntries@restrictedEdit |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntryIds | | Array of ids of the time entries that shall be amended. |
| customerId | | New customer Id that is used to update the time entries. |

AssignTimeEntriesToProjects

With this method, you can assign multiple time entries to a new project.

```
void AssignTimeEntriesToProjects(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] timeEntryIds,  
    Guid projectId,  
    Guid taskId);
```

| Needed Permission | | timeEntries@restrictedEdit |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntryIds | | Array of ids of the time entries that shall be amended. |
| projectId | | New project Id that is used to update the time entries. |
| taskId | | New task Id that is used to update the time entries. |

TIMEPUNCH API SPECIFICATION 3.4

BatchFilterUpdate

This method updates all public and private filters in TimePunch.

```
TPFault BatchFilterUpdate(  
    TPAuthentication authentication,  
    List<TimeEntrySearchDto> privateFilters,  
    List<TimeEntrySearchDto> publicFilters);
```

| Needed Permission | | timeEntryFilter@manage |
|-------------------|----------|-------------------------------------|
| Name | Modifier | Description |
| authentication | | User authentication |
| privateFilters | | Collection of private filters |
| publicFilters | | Collection of public filters |
| Return value | | Fault object if an exception occurs |

CheckTimeEntryOverlapping

This method checks whether the given time entry would override existing time entries in the database. Thus an unintended override of time entries can be prevented.

```
bool CheckTimeEntryOverlapping(  
    out TPFault fault,  
    TPAuthentication authentication,  
    TimeEntryDto entryToCheckOverlapping);
```

| Needed Permission | | timeEntries@access |
|-------------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| entryToCheckOverlapping | | Time Entry that shall be checked against the existing time entries in the database. |
| Return value | | Returns true, if the given time entry would harm existing time entries. |

TIMEPUNCH API SPECIFICATION 3.4

CopyTimeEntries

This method copies a list of time entries and reinstates them on the passed date.

```
void CopyTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<TimeEntrySaveDto> entriesToCopy,  
    DateTime insertAt);
```

| Needed Permission | | timeEntries@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| entriesToCopy | | List of time entries to copy |
| insertAt | | Date from that the time entries shall be inserted. |

CreateBreakTimeDummy

This method creates a break time entry. A break time entry is always required if there is a period of time between the last time entry and the following time entry.

```
TimeEntryDto CreateBreakTimeDummy(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime startBreak,  
    DateTime endBreak);
```

| Needed Permission | | timeEntries@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| startBreak | | Start time of the dummy break entry |
| endBreak | | End time of the dummy break entry. |
| Return value | | The created time entry |

TIMEPUNCH API SPECIFICATION 3.4

CreateNewTimeEntry

This method creates a new time-entry and returns it. The time-entry won't be saved. Only after calling the save method it is available in the TimePunch profile.

```
TimeEntryDto CreateNewTimeEntry(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime date);
```

| Needed Permission | | timeEntries@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| DateTime | | Date that shall be used for creating the new time entry. |
| Return value | | The created time entry |

DeleteTimeEntries

This method deletes the given time entries from the TimePunch profile. It's important to know, that only time-entries of the authenticated TimePunch profile are allowed to be deleted.

```
void DeleteTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] timeEntryIds);
```

| Needed Permission | | timeEntries@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntryIds | | Collection of time entry ids that shall be deleted. |

TIMEPUNCH API SPECIFICATION 3.4

DeleteTimeEntryFilter

This method deletes a time entry filter.

```
TPFault DeleteTimeEntryFilter(  
    TPAAuthentication authentication,  
    Guid filterId);
```

| Needed Permission | | timeEntryFilter@manage timeEntryFilter@managePublic |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| filterId | | Id of the filter that shall be deleted |
| Return value | | Fault object if an exception occurs |

ImportTimeEntries

This method imports the transferred time entries. The import takes into account not the IDs but the start and end times, as well as the project and activity names.

```
void ImportTimeEntries(  
    out TPFault fault,  
    TPAAuthentication authentication,  
    List<TimeEntryDto> rows);
```

| Needed Permission | | timeEntries@restrictedEdit commonTasks@manage projects@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| rows | | Collection of time entries to import |

TIMEPUNCH API SPECIFICATION 3.4

LoadTimeEntries

This method should be used, if the time entry ids are known, in order to load the time entries.

```
List<TimeEntryDto> LoadTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] timeEntryIds);
```

| Needed Permission | | timeEntries@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntryIds | | Array of ids of the time entries that shall be loaded. |
| Return value | | Returns the loaded time entries |

LoadTimeEntryFilter

This method loads a time entry filter with the corresponding ID.

```
TimeEntrySearchDto LoadTimeEntryFilter(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid timeEntrySearchId);
```

| Needed Permission | | timeEntryFilter@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntrySearchId | | Id of the time entry filter |
| Return value | | Returns the loaded time entry filter |

TIMEPUNCH API SPECIFICATION 3.4

MarkTimeEntries

This method marks the transferred time entries with the corresponding marker in TimePunch.

```
void MarkTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<Guid> selectedIds,  
    TimeEntryMarkAs markAs)
```

| Needed Permission | | timeEntries@restrictedEdit |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| selectedIds | | List of time entry ids to mark |
| markAs | | Marker that shall be set for the time entries Paid = 1 Important = 2 Onsite = 4 |

SaveTimeEntryFilter

This method saves a time entry filter in the database. The mapping is not carried out using the ID, but the name of the report filter.

```
TimeEntrySearchDto SaveTimeEntryFilter(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySearchDto timeEntryFilter);
```

| Needed Permission | | timeEntryFilter@manage timeEntryFilter@managePublic |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntryFilter | | Time Entry Filter so save |
| Return value | | The stored time entry filter |

TIMEPUNCH API SPECIFICATION 3.4

SearchTimeEntries

This method searches for time-entries with the given filter. The permission of the authenticated user will be considered. Only time entries of the user profiles that the user is allowed to access will be returned.

```
List<TimeEntryDto> SearchTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySearchDto search);
```

| Needed Permission | | timeEntries@access / core@switchIdentity |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| search | | The filter that is used to search the time entries |
| Return value | | Collection with the found time entries |

SearchTimeEntryFilters

This method uses the search filters criteria to find time entry filters in the database. A list of all the time entry filters found is then returned.

```
List<TimeEntrySearchDto> SearchTimeEntryFilters(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntryFilterSearchDto searchFilterCriteria);
```

| Needed Permission | | timeEntryFilter@access |
|----------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| searchFilterCriteria | | Searchfilter criteria |
| Return value | | Collection containing the found time entry search objects |

TIMEPUNCH API SPECIFICATION 3.4

SearchAssignableUsers

This method returns all employee profiles that can be assigned to the transferred group type.

```
UserGroupMemberDto[] SearchAssignableUsers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    GroupType groupType,  
    Guid userGroupId);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| groupType | | Group type for that the user profiles shall be searched |
| userGroupId | | Id of the user group that shall be used as a target for the assignable user profiles. |
| Return value | | Returns the found user groups |

SearchCoveredUserGroups

This method searches for user groups in which the employee is either in charge or at least entered as an assistant.

```
UserGroupDto[] SearchCoveredUserGroups(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserGroupSearchDto userGroupSearch);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userGroupSearch | | The user group definition to search for |
| Return value | | Returns the found user groups |

TIMEPUNCH API SPECIFICATION 3.4

UnmarkTimeEntries

This method removes the markers on the transferred time entries.

```
void UnmarkTimeEntries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<Guid> selectedIds,  
    TimeEntryMarkAs markAs);
```

| Needed Permission | | timeEntries@restrictedEdit |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| selectedIds | | List of time entry ids to mark |
| markAs | | Marker that shall be removed from the time entries Paid = 1 Important = 2 Onsite = 4 |

UpdateTimeEntryFilter

This method updates an existing time entry filter. To call the update method, the time entry filter to be updated should already exist.

```
TimeEntrySearchDto UpdateTimeEntryFilter(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySearchDto filter);
```

| Needed Permission | | timeEntryFilter@manage timeEntryFilter@managePublic |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| filter | | Time Entry Filter so save |
| Return value | | The stored time entry filter |

TIMEPUNCH API SPECIFICATION 3.4

ValidateAndSaveBreakReplacement

This method was specially developed to be used by employees without special rights. It is used so that existing breaks, that is, the time between two time entries, can be assigned with a project by an employee without special rights.

```
TimeEntrySaveResultDto ValidateAndSaveBreakReplacement(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySaveDto timeEntry);
```

| Needed Permission | | timeEntries@restrictedEdit |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntry | | Time entry that shall be stored in the database. The time entry must fit into a break time. |
| Return value | | Returns the result of storing the time entry (Ok, BreakEnforced, EmptyBreakEntry) and the stored time entries |

ValidateAndSaveRestrictedTimeEntry

This method is used to amend a time entry. Because the method needs less permission, one can only change the description, the project and the task of the time-entry.

```
TimeEntrySaveResultDto ValidateAndSaveRestrictedTimeEntry(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntryRestrictedSaveDto timeEntry);
```

| Needed Permission | | timeEntries@restrictedEdit |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntry | | Time entry that shall be stored in the database. |
| Return value | | Returns the result of storing the time entry (Ok, BreakEnforced, EmptyBreakEntry) |

TIMEPUNCH API SPECIFICATION 3.4

ValidateAndSaveTimeEntry

This method takes a time entry, validates it and saves it in the database.

```
TimeEntrySaveResultDto ValidateAndSaveTimeEntry(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySaveDto timeEntry);
```

| Needed Permission | | timeEntries@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntry | | Time entry that shall be stored in the database. |
| Return value | | Returns the result of storing the time entry (Ok, BreakEnforced, EmptyBreakEntry) |

ValidateAndSaveTimeEntrySeries

This method is used to create a time entry series, e.g. in order to create holiday entries. For that it's necessary to define the start and end date. The concrete times are taken from the working times model of the user.

```
TimeEntrySaveResultDto ValidateAndSaveTimeEntrySeries(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySeriesDto timeEntrySeries);
```

| Needed Permission | | timeEntries@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntrySeries | | Time entry series that shall be stored in the database. |
| Return value | | Returns the result of storing the time entry (Ok, BreakEnforced, EmptyBreakEntry) and the stored time entries |

TIMEPUNCH API SPECIFICATION 3.4

ValidateAndSaveTimeEntryCopies

Using this method copies of time entries can be created, e.g. to create an entry that occurs on several day in a row. Maybe the staff member takes part on a weekly trainee.

```
TimeEntrySaveResultDto ValidateAndSaveTimeEntryCopies(  
    out TpFault fault,  
    TpAuthentication authentication,  
    TimeEntrySaveDto timeEntry);
```

| Needed Permission | | timeEntries@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| timeEntry | | Time entry copy that shall be stored in the database. |
| Return value | | Returns the result of storing the time entry (Ok, BreakEnforced, EmptyBreakEntry) and the stored time entries |

TimePunch User Group Service

The user group service allows grouping of employee profiles. Thus, employees can be grouped together for teams, departments, and offices.

CreateUserGroup

This method creates a new group and returns it. It is important that the group not yet gets stored to the database. This happens only when the group is saved.

```
UserGroupDto CreateUserGroup(  
    out TpFault fault,  
    TpAuthentication authentication,  
    GroupType groupType);
```

| Needed Permission | | userGroups@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| groupType | | Type of the user group that shall be created (e.g. Team, Department, Branch) |
| Return value | | Returns the created user group |

DeleteUserGroups

This method clears the passed user groups. The members of the group will not be associated any other group.

```
void DeleteUserGroups(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] userGroupIds);
```

| Needed Permission | | userGroups@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userGroupIds | | List that contains all user group ids that shall be deleted. |
| Return value | | - |

ImportUserGroupAssignments

This method imports the passed group memberships. In contrast to the save method, the ID and not the common name of the group is important.

```
UserGroupMemberDto[] ImportUserGroupAssignments(  
    out TpFault fault,  
    TpAuthentication authentication,  
    GroupType groupType,  
    UserGroupMemberDto[] groupAssignments);
```

| Needed Permission | | userGroups@import |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| groupType | | Type of the user group for which the group assignments shall be imported |
| groupAssignments | | User Group assignments to import |
| Return value | | - |

LoadUserGroup

This method loads the group with the specified ID and returns it.

```
UserGroupDto LoadUserGroup(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid userGroupId);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userGroupId | | Id of the user group to load |
| Return value | | Returns the loaded user group |

SaveUserGroup

This method saves the user group in the database. It is important that if the group members in the object are NULL, only the basic data of the Group gets stored.

```
UserGroupDto SaveUserGroup(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserGroupDto userGroup);
```

| Needed Permission | | userGroups@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userGroup | | The user group to save |
| Return value | | Returns the persisted user group |

SaveUserGroupAssignments

This method saves the group membership of employee profiles to the given group.

```
void SaveUserGroupAssignments(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserGroupSaveMemberDto[] groupAssignments);
```

| Needed Permission | | userGroups@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| groupAssignments | | User Group assignments to save |
| Return value | | - |

TIMEPUNCH API SPECIFICATION 3.4

SearchAssignableUsers

This method returns all employee profiles that can be associated with the passed type of group.

```
UserGroupMemberDto[] SearchAssignableUsers(  
    out TpFault fault,  
    TpAuthentication authentication,  
    GroupType groupType,  
    Guid userGroupId);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| groupType | | Group type for that the user profiles shall be searched |
| userGroupId | | Id of the user group that shall be used as a target for the assignable user profiles. |
| Return value | | Returns the found user groups |

SearchCoveredUserGroups

This method searches for user groups where the employee has the lead, or is registered at least as an Assistant of the group.

```
UserGroupDto[] SearchCoveredUserGroups(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserGroupSearchDto userGroupSearch);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userGroupSearch | | The user group definition to search for |
| Return value | | Returns the found user groups |

SearchUserGroupAssignments

This method returns all members of the group to which the profile search fits and agrees with the passed type of group. The data are used primarily for the export of the group.

```
UserGroupMemberDto[] SearchUserGroupAssignments(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto userSearch,  
    GroupType groupType);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSearch | | Use the given user search object to tailor the search results. |
| groupType | | Group type for that the user profiles shall be searched |
| Return value | | Returns the found user assignments |

SearchUserGroup

This method searches for user groups with the given properties. It can be searched for the group name and the group type.

```
UserGroupDto[] SearchUserGroups(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserGroupSearchDto userGroupSearch);
```

| Needed Permission | | userGroups@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userGroupSearch | | The user group definition to search for |
| Return value | | Returns the found user groups |

TimePunch User Profile Service

This service provides methods to manipulate staff profiles in TimePunch.

CreateNewUserProfile

Using this method, you can create a new employee profile in TimePunch. This requires only a few details. As a result, the core permissions will be returned of the employee.

```
UserProfileRightsDto CreateNewUserProfile(  
    out TpFault fault,  
    TpAuthentication authentication,  
    string userName,  
    Guid? copyFromUserId,  
    string[] applicationKeys,  
    bool directLogon,  
    out List<string> permissions);
```

| Needed Permission | | --- |
|------------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication with proposed logon and password information |
| userName | | Fore -and sure name of the user that shall be created |
| copyFromUserId | | NULL, if no defaults shall be copied or a valid user id, if the e.g. the worktime settings shall be copied. |
| applicationKeys | | The applications for that the user shall be created. |
| directLogon | | True, if the user shall be directly licensed. False, if the user shall only be created, but need not to be registered directly. |
| Permissions | Out | List that contains the permissions of the newly created user. |
| return value | | Object that contains the core user data and the permissions. |

DeleteUserProfileDataOlderThan

You can use this method to delete time data from existing user profiles. For this purpose, the delete method is given a period of years. Time data that is older than the specified number of years is deleted from the transferred profiles.

```
void DeleteUserProfileDataOlderThan(  
    out TpFault fault,  
    TpAuthentication authentication,  
    int olderThanYears,  
    Guid[] userIds);
```

Note: Only fully completed years are considered.

This means that if data older than 1 year is to be deleted in May 2018, the data will be deleted by 31.12.2016.

| Needed Permission | | dsgvo@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| olderThanYears | | Specifies the amount of years to look back in time |
| userIds | | User ids for that the data shall be deleted. |

DownloadProfileImages

This method loads the profile images of specified TimePunch profiles.

```
List<UserProfileImageDto> DownloadProfileImages(  
    out TpFault fault,  
    TpAuthentication authentication,  
    ImageFormat imageFormat,  
    Guid[] userIds);
```

| Needed Permission | | --- |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| imageFormat | | The image format (e.g. max size) the user profiles shall be returned. Valid values are: Thumbnail16, Thumbnail24, Thumbnail32, Thumbnail64, Thumbnail128, Thumbnail256, Thumbnail512, Portrait300X400, Portrait512X682, Portrait600X800, Portrait768X1024, Portrait900X1200 |
| Return value | | List containing the profile images for the given profiles. |

TIMEPUNCH API SPECIFICATION 3.4

ExportUserProfileData

This method exports the profile data of selected employees. The search is used for this purpose.

```
List<UserProfileDataDto> ExportUserProfileData(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserSearchDto search);
```

| Needed Permission | | userProfiles@export |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| UserSearchDto | | The search object that is used to find the profiles to export. |
| Return value | | List containing the profile data for the found profiles. |

ImportUserProfileData

This method imports the profile data of the passed-employees. The logon is crucial for the creation or update of the profiles.

```
List<UserProfileDataDto> ImportUserProfileData(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<UserProfileDataDto> userProfiles);
```

| Needed Permission | | userProfiles@import |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userProfiles | | List containing the profile data that shall be imported. |
| Return value | | List containing the profile data that has been imported. |

LoadUserProfileDefaults

The method returns the default values for the specified employee profiles. Those values serve the internal logic of the calculation (flexible working hours / holidays) as offset or underlying.

```
List<UserProfileDefaultsDto> LoadUserProfileDefaults(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] userIds);
```

| Needed Permission | | userProfiles@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userIds | | The user ids of the members from which the profile defaults shall be loaded |
| Return value | | List with default profile information of the specified users. |

LoadUserProfileDetails

This method loads the detailed information about the employee profile.

```
List<UserProfileDetailsDto> LoadUserProfileDetails(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] userIds);
```

| Needed Permission | | userProfiles@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userIds | | The user ids of the members from which the profile details shall be loaded |
| Return value | | List with detailed profile information of the specified users. |

LoadUserProfileRights

This method loads the permissions of the specified employee profiles.

```
List<UserProfileRightsDto> LoadUserProfileRights(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] userIds);
```

| Needed Permission | | permissions@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userIds | | The user ids of the members from which the permissions shall be loaded |
| Return value | | List with profile permission information of the specified users. |

LoadUserProfileSensitive

This method loads relevant and sensitive employee data from the transferred employee profiles. The data can only be read out by users with the appropriate authorization and only if the employees are present in the user's coverage.

```
List<UserProfileSensitiveDto> LoadUserProfileSensitive(  
    out TpFault fault,  
    TpAuthentication authentication,  
    Guid[] userIds)
```

| Needed Permission | | userSensitive@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userIds | | The user ids of the members from which the sensitive data shall be loaded |
| Return value | | List with sensitive information of the specified users. |

SaveUserProfileDefaults

This method updates the default values for the specified employee profiles. This works only, if still no monthly closing has been done for the profiles. The SaveUserProfileDetails method must be used to update basic data.

```
List<UserProfileDefaultsDto> SaveUserProfileDefaults(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<UserProfileDefaultsDto> defaults);
```

| Needed Permission | | userProfiles@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| details | | List containing the profile defaults that shall be updated. |
| Return value | | List containing the profile defaults that has been updated. |

SaveUserProfileDetails

This method updates the profile information for one or more employees.

```
List<UserProfileDetailsDto> SaveUserProfileDetails(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<UserProfileDetailsDto> details);
```

| Needed Permission | | userProfiles@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| details | | List containing the profile data that shall be updated. |
| Return value | | List containing the profile data that has been updated. |

TIMEPUNCH API SPECIFICATION 3.4

SaveUserProfileRights

This method updates the rights of the specified employee profiles. The SaveUserProfileDetails method must be used to update basic data.

```
List<UserProfileRightsDto> SaveUserProfileRights(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<UserProfileRightsDto> rights);
```

| Needed Permission | | permissions@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| details | | List containing the profile rights that shall be updated. |
| Return value | | List containing the profile rights that has been updated. |

SaveUserProfileSensitive

This method stores employee information relevant to data protection. The data can only be stored by users with appropriate authorization and only if the employees are present in the user's coverage.

```
List<UserProfileSensitiveDto> SaveUserProfileSensitive(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<UserProfileSensitiveDto> sensitive);
```

| Needed Permission | | userSensitive@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| sensitive | | List containing the sensitive user data that shall be updated. |
| Return value | | List containing the profile rights that has been updated. |

SearchUserProfilesToDelete

This method searches for user profiles that contain data that is older than the specified time in years. The profiles found can serve as a basis for the method `DeleteUserProfileDataOlderThan`.

```
List<UserDSGVOInfoDto> SearchUserProfilesToDelete(  
    out TpFault fault,  
    TpAuthentication authentication,  
    int olderThanYears,  
    UserSearchDto userSearchDto);
```

| Needed Permission | | dsgvo@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| olderThanYears | | Specifies the amount of years to look back in time |
| userSearchDto | | Search definition used to search the user profiles |

UploadProfileImage

This method updates the profile pictures for the given employee profiles.

```
void UploadProfileImage(  
    out TpFault fault,  
    TpAuthentication authentication,  
    UserProfileImageDto[] profileImages);
```

| Needed Permission | | userProfiles@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| profileImages | | List containing the profile images that shall be updated. |

TimePunch Work Model Service

The work model service offers methods to access the work models of the staff members.

AddEmptyWorkmodel

This method is used to add an empty working time model to an employee who leaves the company. From this date on, the employee has officially no longer any planned working time. This freezes the current overtime.

```
void AddEmptyWorkmodel(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime dateOfLeaving);
```

| Needed Permission | | workdayModels@manage |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| dateOfLeaving | | Date used to evaluate add a new empty workmodel |

GetWorkDayModels

Using this method the workday models for the given user can be retrieved starting by start date through the end date.

```
List<WorkdayDto> GetWorkDayModels(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime startDate,  
    DateTime endDate);
```

| Needed Permission | | workdayModels@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| userSearch | | Information that are used to search the TimePunch profiles |
| startDate | | First date for that the workday model shall be retrieved. |
| endDate | | Last date for that the workday model shall be retrieved. |
| Return value | | List of all workday models for the given authentication. |

TIMEPUNCH API SPECIFICATION 3.4

GetWorkModel

This method returns the working time model for the passed date.

```
WorkModelDto GetWorkModel(  
    out TpFault fault,  
    TpAuthentication authentication,  
    DateTime date,  
    Guid? userId = null);
```

| Needed Permission | | workdayModels@access |
|-------------------|----------|---|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| date | | Date used to evaluate the valid work model |
| userId | | User Id for whom the work model shall be received. Or NULL to evaluate it for the current user. |
| Return value | | Work model that is valid for the given date. |

GetWorkModels

This method returns all working time models for the given employee.

```
List<WorkModelDto> GetWorkModels(  
    out TpFault fault,  
    TpAuthentication authentication);
```

| Needed Permission | | workdayModels@access |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| Return value | | List of all work models for the authenticated identity |

TIMEPUNCH API SPECIFICATION 3.4

SaveWorkModels

This method saves the work schedules for the given employee. It is important that the list is complete, missing working-time models in the staff profile are getting removed.

```
void SaveWorkModels(  
    out TpFault fault,  
    TpAuthentication authentication,  
    List<WorkModelDto> workModels);
```

| Needed Permission | | workdayModels@manage |
|-------------------|----------|--|
| Name | Modifier | Description |
| fault | Out | Contains the error if an exception occurs. |
| authentication | | User authentication |
| workModels | | List of all work models for the authenticated identity |

Entities

This chapter contains information about the entities and data structures of the TimePunch API.

Core

TpFault

The object of the class TpFault will always be returned if an error occurs.

| Datatype | Name | Description |
|----------|------------|--------------------|
| Integer | ErrorCode | Error Code Nr. |
| String | Message | Error Message |
| String | FaultType | Exception Type |
| TpFault | InnerFault | Inner fault object |

TpAuthenticationFault : TpFault

Die Klasse *TpAuthenticationFault* ist von *TpFault* abgeleitet. Die Klasse hat ein weiteres Feld mit dem Namen *Failure* welches den Grund für den Autorisierungsfehler enthält.

| Datatype | Name | Description |
|-----------------------|---------|--|
| AuthenticationFailure | Failure | <div>UnexpectedException ⇒ an unexpected exception occurred</div> <div>NoRemoteConnection ⇒ Not used exception</div> <div>NoDatabaseConnection ⇒ There's no database connection defined</div> <div>UserProfileNotFound ⇒ The given user profile couldn't be found</div> <div>IncorrectPassword ⇒ The password wasn't correct</div> <div>IdentityProfileNotFound ⇒ The given identity couldn't be found</div> <div>PasswordRequired ⇒ No password has been set</div> <div>OldDatabaseVersion ⇒ The database structure is outdated must be updated</div> |

TIMEPUNCH API SPECIFICATION 3.4

TimePunchOutdated

- ⇒ The database structure is newer than the running TimePunch version. So TimePunch needs to be updated.

NoLogonName

- ⇒ No user logon has been set

DataBelongsToDifferentProfile

- ⇒ The user tries to edit data which belongs to a different user profile.

NotInCoverage

- ⇒ The user tries to edit data of a user profile that is not in his/her coverage.

InsufficientRights

- ⇒ The user has no permission to execute the command.

InsufficientRightsToMigrateDatabase

- ⇒ The user tries to migrate the database to a newer version. This is only allowed for people who have the rights to administrate TimePunch.

PagingContextDto

This object is the base class for all search object. It includes an option for paging the results

| Datatype | Name | Description |
|----------|-----------|---|
| Bool | UsePaging | Yes, if paging shall be used. No, if all result shall be returned at once. |
| Int | Page | Zero based number of the page, or empty if no paging is used. |
| Int | PageSize | Number of entries at the page, or empty if no paging is used. |

TimePunch Authentication Service

TpAuthentication

The authentication object has to be send with every method whenever a user-authentication is required. This ensures that the user has the required access rights.

| Datatype | Name | Description |
|----------|---------------|---|
| String | CustomerToken | Not used at the moment. |
| String | TpUser | TimePunch Profile of the used Principal |
| String | TpHashedPwd | Password Hash (build with MD5 Encryption) |
| String | Identity | Profile Name of the used Identity, or empty, if the identity is equal to the principal. In order to switch the identity, the principal needs administration rights. |
| String | Culture | Culture that is used for the localization. e.g. „de-DE“ |

UserProfileDto

Diese Klasse enthält die Basisdaten um ein TimePunch Profil zu identifizieren.

| Datatype | Name | Description |
|----------|------------|---|
| Guid | Id | Unique ID of TimePunch Profile |
| String | SaveAsName | Name of the TimePunch Profile used to show the user. |
| String | LogonName | By default the logon name is identical to the used windows logon. |
| String | UserName | First name and last name of the user. |
| String | Email | Email of the user |

TIMEPUNCH API SPECIFICATION 3.4

UserLogonDto : UserProfileDto

This class is derived from the UserProfileDto and contains the base data and in addition a TimePunch profile to identify expanded information.

| Datatype | Name | Description |
|------------|---------------------|--|
| UserRights | Rights | Permission of the user |
| DateTime? | FirstEntry | Date/Time of the first entry, if the user has an entry, otherwise NULL |
| DateTime? | LastEntry | Date/Time of the last entry, if the user has an entry, otherwise NULL |
| Bool | IsUserProfileActive | True, if the user profile is active. |
| DateTime? | LockedUpTo | Date until that the time entries are locked. NULL, if no time entries are locked at all. |
| Bool | IsInCoverage | True, if the user profile is in coverage |

UserSearchDto : PagingContextDto

This class contains all basic data of PagingContextDto and defines a filter to search for TimePunch profiles.

| Datatype | Name | Description |
|---------------|---------------|---|
| UserFiltering | UserFiltering | Defines if the users are filtered (CurrentUser, AllUser, SelfDefined) |
| Guid[] | UserIds | If Filter is set to "SelfDefined" than the users will be filtered by the given ids. |
| String | GenericSearch | Generic search string that search for logon and/or user name. |
| Bool? | IsUserDeleted | NULL, searches for deleted and active users. True, to search only for deleted users. False, to search only for non-deleted users. |

AutoregistrationDto

This class contains information about whether automatic registration of new employees is allowed.

| Datatype | Name | Description |
|------------|---------------------------|---|
| Bool | IsAutoregistrationAllowed | Defines whether the auto-registration of new staff members is allowed. |
| UserRights | DefaultUserRights | Get the user rights of new staff members |
| Bool | IsAutomaticLicensing | Defines whether new members will get a new license. But only if there are licenses still available. |

TimePunch Configuration Service

CostCenterDto

This class contains information about the cost center assignment of the different booking types in TimePunch.

| Datatype | Name | Description |
|----------|-------------------------------|---|
| String | SicknessCostcenter | Cost center that is used for sickness bookings |
| String | LeaveCostcenter | Cost center that is used for leave bookings |
| String | OfficialBankHolidayCostcenter | Cost center that is used for official bank holiday bookings |
| String | SpecialLeaveCostCenter | Cost center that is used for special leave bookings |
| String | CorrectionCostCenter | Cost center that is used for overtime/flexitime corrections |

SalaryExportDto

This class contains information about the settings for wage data export.

| Datatype | Name | Description |
|----------|---------------------------|--|
| String | FixedSalaryTypeNo | Defines the account for the fixed salary export. |
| String | TemporaryHourlyWageTypeNo | Defines the account for the temporary hourly wage workers. |
| String | HourlyWageTypeNo | Defines the account for the hourly wage workers. |
| String | SicknessTypeNo | Defines the account for the time of sickness. |
| String | LeaveTypeNo | Defines the account for the leave time. |
| String | SpecialLeaveTypeNo | Defines the account for the special leave time. |
| String | PaidOvertimeTypeNo | Defines the account for the paid overtime. |
| String | OfficialBankHolidayTypeNo | Defines the account for the official bank holiday. |
| String | SaturdayWorkAmTypeNo | Defines the account for the work at Saturday morning. |
| String | SaturdayWorkPmTypeNo | Defines the account for the work at Saturday afternoon. |
| String | SundayWorkTypeNo | Defines the account for the worktime at Sunday. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|------------------|----------------------|--|
| String | NightShiftTypeNo | Defines the account for the night shift working time. |
| String | NightShiftCoreTypeNo | Defines the account for the night shift core working time. |
| SalaryExportType | ExportType | <p>Defines the type of the salary export.</p> <p>SimplePayout ⇒ The complete month gets exported</p> <p>SubsequentPayout ⇒ Export in the middle of the month using DATEV retroactive billing</p> <p>FloatingPayout ⇒ Export in the middle of the month without using the DATEV recalculation</p> |

DatevExportSaveDto: SalaryExportDto

This class contains information about the settings for exporting to DATEV..

| Datatype | Name | Description |
|----------|--------------|--|
| String | AccountendNo | Defines the number of the accountend. This is the company who manages the DATEV access for the customer. |
| String | ClientNo | Defines the client number of the customer. |

DatevExportDto: DatevExportSaveDto

In addition to the actual DATEV information, this class also contains data on the set cost centers.

| Datatype | Name | Description |
|----------|-------------------------------|---|
| String | SicknessCostcenter | Cost center that is used for sickness bookings |
| String | LeaveCostcenter | Cost center that is used for leave bookings |
| String | OfficialBankHolidayCostcenter | Cost center that is used for official bank holiday bookings |
| String | SpecialLeaveCostCenter | Cost center that is used for special leave bookings |
| String | CorrectionCostCenter | Cost center that is used for overtime/flexitime corrections |

TIMEPUNCH API SPECIFICATION 3.4

LexwareExportDto: SalaryExportDto

In addition to the actual wage information, this class also contains data on the set cost centers.

UserSettingsDto

This class contains settings for the behaviour of TimePunch.

| Datatype | Name | Description |
|---------------|--------------------------------|--|
| Bool | IsAllowedToOverwriteBreaks | Flag that indicates whether a break can be overwritten by user with restricted access. This value is false by default, because we don't want the user (without the managing permissions) to do that. |
| String | IsAllowedToManageProjectAccess | Flat that indicates whether the timepunch user can manage project access. This value is true by default. It may only be false, if the project access gets managed by a 3rd party software |

TimePunch Customer Service

CustomerDto

This class contains all customer data.

| Datatype | Name | Description |
|---------------------------------------|----------------------------------|--|
| Guid | Id | Unique ID of the customer |
| String | CustomerName | Name of the customer. This field, combined with the CustomerRefNr is unique. |
| String | CustomerRefNr | Ref. Nr. of the customer. This field, combined with the CustomerName is unique. |
| String | CustomerCode | Barcode of a customer. Used to identify a customer by its barcode. |
| String | EEmailAddress | Email address of the customer |
| String | PostalAddress | Postal address of the customer. |
| String | Website | Customers web site URL |
| String | ContactPerson | Name of the contact person of the customer |
| String | ContactPerson JobPosition | Job Position of the contact person |
| String | PhoneNumber | The phone number of the customer or/and the contact person. |
| String | FaxNumber | The fax number of the customer or/and the contact person. |
| String | MobileNumber | The mobile number of the customer or/and the contact person. |
| String | Notes | Notes field to describe the customer |
| Datatype | Name | Description |
| ColumnFlagsFor TimeSheetExport | ColumnFlags | <p>A bit field that defines the columns that are used for exporting data to the customer.</p> <pre> None = 1, Date = 2, Weekday = 4, Project = 16, Task = 32, StartTime = 64, WorkingTime = 256, 512, Flexitime = 2048, Journey = 8192, None = 1, Date = 2, Weekday = 4, Project = 16, Task = 32, StartTime = 64, WorkingTime = 256, 512, Flexitime = 2048, Journey = 8192, Description = 8, EndTime = 128, BreakTime = 1024, Sick = 4096, HourlyRate = 16384 </pre> |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|--------------------|-----------------------------|---|
| String | ExportTimeSheetMailTemplate | Mail template that is used to send the time sheet to the customer. |
| SyncTimeSheetFlags | SyncSheetFlags | A bit field that defines if also the time sheet data will be send to the customer (wich is the receiver)/ or a second sender (sender). Undefined = 0, None = 1, SyncToSender = 2, SyncToReceiver = 4 |
| Bool | IsCustomerDeleted | True, if the customer has been marked as deleted. |

CustomerSearchDto

This class is derived from the PagingContextDto and contains the fields that can be used to search for customers.

| Datatype | Name | Description |
|----------|--------------------------|---|
| String | CustomerName | Name of the customer. This field, combined with the CustomerRefNr is unique. |
| String | CustomerRefNr | Ref. Nr. of the customer. This field, combined with the CustomerName is unique. |
| String | EEmailAddress | Email address of the customer |
| String | PostalAddress | Postal address of the customer. |
| String | Website | Customers web site URL |
| String | ContactPerson | Name of the contact person of the customer |
| String | ContactPersonJobPosition | Job Position of the contact person |
| String | PhoneNumber | The phone number of the customer or/and the contact person. |
| String | FaxNumber | The fax number of the customer or/and the contact person. |
| String | MobileNumber | The mobile number of the customer or/and the contact person. |
| String | Notes | Notes field to describe the customer |
| Datatype | Name | Description |
| String | GenericSearch | The generic search will search for customers by the name and also the ref. Nr. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|-------------------------|---------------------------------|---|
| String | ExportTimeSheet MailTemplate | Mail template that is used to send the time sheet to the customer. |
| List<Guid> | CustomerIds | The method will only search in the given customer ids. |
| bool? | IsCustomerDeleted | NULL to search all customers independent of the state / True, False to search customers with a defined state. |

TimePunch Holiday Service

CountryDto

This class contains the definition of a country.

| Datatype | Name | Description |
|-------------|---------|---|
| Guid | Id | Unique ID of the country |
| String | Name | Name of the country, e.g. Deutschland |
| RegionDto[] | Regions | List of Regions that belongs to the country (e.g. Hessen) |

RegionDto

This class contains information about a region (for example, Hessen) and is embedded in a CountryDTO object.

| Datatype | Name | Description |
|----------|------|---------------------|
| Guid | Id | Unique ID of Region |
| String | Name | Name of the Region |

PublicHolidayDto

This class contains information about a public holiday.

| Datatype | Name | Description |
|-------------------|-------------------|---|
| Guid | CountryId | Unique ID of the Country |
| String | Country | Name of the country |
| Guid? | RegionId | Unique ID of the Region |
| String | Region | Name of the region |
| String | Name | Name of the public holiday |
| Int? | ValidSinceYear | Year since the holiday is valid |
| Int? | ValidUntilYear | Year until the holiday is valid |
| Bool | IsHalfDay | True, if it's only a half day holiday |
| PublicHolidayType | PublicHolidayType | Type of the public holiday |
| DateTime | Date | Date of the public holiday |
| Bool | Reverse | True, indicates that the numbered holiday will be reversed. |
| Int | Number | Number that is used for counting when used in numbered holidays |
| DayOfWeek | Weekday | Weekday of the public holiday |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|-------------------------|-----------------------|--|
| PublicHolidayDto | ReferencedHoliday | Referenced holiday if the current holiday is dependend on it |
| string | ValidationInformation | Gets a string that defines when the holiday is valid |

TimePunch Licensing Service

ApplicationLicenseDto

This class contains the license data of TimePunch.

| Datatype | Name | Description |
|----------|-----------------|---|
| Guid | Id | Unique ID of the application license |
| String | LicensedFor | Name of whom the license has been registered. |
| String | LicenseCode | License Code |
| Int | CountKeys | Amount of users who can use this application license. |
| Int | UsedKeys | Amount of licenses that are already in use. |
| DateTime | ValidUntil | If the license is only valid for a time period, this field contains the expiration date. |
| String | ValidForVersion | If the license is only valid for a given TimePunch Version, this version is filled in here. |
| Bool | IsLicenseValid | True, if the returned application license is valid for the installed version of TimePunch. |
| String | ApplicationKey | The application key to which the application license belongs. |

UserLicenseDto

This class contains the license information for a TimePunch profile.

| Datatype | Name | Description |
|----------------|----------------------|---|
| Guid | Id | Unique ID of the application license |
| UserProfileDto | User | TimePunch Profile to that the user license belongs too |
| String | ApplicationKey | The module or application key for that the license is valid |
| Guid | ApplicationLicenseId | The Id of the application license for which the user is licensed. |

TimePunch Logging Service

ActiveTimeEntryDto

This class contains the data of the currently active time-entry. The active time entry is the one for which the current time is recorded.

| Datatype | Name | Description |
|------------------------|-----------------|---|
| Guid | Id | Unique ID of the active time entry |
| DateTime | LastUpdate | Time stamp of the last update |
| Guid | ProjectId | Unique project Id for that the time tracking has been started. |
| String | ProjectName | Name of the project for that the time tracking has been started. |
| Guid | TaskId | Unique Task Id, if the time tracking has been started for a specific task. |
| String | TaskName | Name of the task, if the time tracking has been started for a specific task. |
| String | Description | Description of the current time entry. |
| DateTime | LogonTime | Start time of the current time tracking. |
| Double | BreakTime | Amount of break time in hours within the current time tracking. |
| DateTime | BreakStartTime | Start time of the current break, if the user started a break. |
| DateTime | BreakEndTime | End time of the last break, if the break has been finished. |
| TimeEntry-LoggingState | LoggingState | Defines the state of the current logging (NoActiveEntry, WorkTimeLogging, BreakTimeLogging) |
| Long | BackgroundColor | Background colour of the project that gets tracked. |
| Long | TextColor | Text colour of the project that gets tracked. |
| DateTime | LockTime | Timestamp where the PC got locked due to a screensaver event etc. |
| String | LogonName | Dependent TimePunch profile |
| Datatype | Name | Description |
| Guid | UserId | Id of the user dependent TimePunch Profile |
| String | CustomerName | Name of the customer that has been set for the time entry. |
| String | CustomerRefNr | Ref.Nr. of the customer that has been set for the time entry |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|---------------|--------------------|---|
| Guid | CustomerId | Unique customer Id |
| Bool | IsImportant | True, if the time entry has been marked as important. This will force the project report to show the description. |
| Bool | IsOnsite | True, if the time entry has been marked as onsite. |
| String | ProjectDescription | Gets the project description |
| String | TaskNameId | Gets the displayed task id |
| String | TaskDescription | Gets the task description |
| TimeEntryType | Usage | Gets the usage of the active time entry |
| String | TimeZone | Gets the timezone where the active time entry belongs too |
| Bool | IsNotInvoiced | True, if the active time entry gets not invoiced at all. |

ActiveTimeEntrySearchDto

This class is derived from the PagingContextDto and will be used to search for active time entries.

| Datatype | Name | Description |
|------------|------------------|---|
| List<Guid> | FilteredProjects | If null, all projects will be returned. If the collection is empty, nothing will be returned. If the collection is filled with project ids, only entries with the given project ids will be returned. |
| List<Guid> | FilteredTasks | If null, all tasks will be returned. If the collection is empty, nothing will be returned. If the collection is filled with task ids, only entries with the given task ids will be returned. |
| List<Guid> | FilteredUsers | If null or empty, the data of the current user will be returned. If the collection is filled with, only entries with the given user ids will be returned. |

LoggingContextDto

The Logging Context contains all data that is necessary for the client in order to do a proper time-recording.

| Datatype | Name | Description |
|------------------|----------------|---|
| List<ProjectDto> | ActiveProjects | Collection containing all projects the user can book times. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|--------------------|-------------------|---|
| List<TaskDto> | CommonTasks | Collection of common tasks |
| ActiveTimeEntryDto | ActiveTimeEntry | Active time entry, if a logging has been started. |
| LoggingSettingsDto | Settings | Logging Settings for the current user |
| List<TimeEntryDto> | TodayEntries | Collection of time entries of the current day |
| Guid | ProposedProjectId | Id of the Project that should be used to start the next time recording. |
| Guid | ProposedTaskId | Id of the Task that should be used with the next time recording. |
| TimeAccountDto | TimeAccount | Current time account information. |
| List<CustomerDto> | Customers | List of customers the user can choose from. |

LoggingSettingsDto

This class contains the recording settings for the TimePunch profile.

| Datatype | Name | Description |
|---------------------|-------------------------|---|
| Bool | InstantLogging | Value indicating whether the time recording shall start directly after starting the TimePunch Watcher. |
| Recording-Mode | RecordingMode | Value indicating in which recording mode the TimePunch Watcher shall be used (Complex, Simple, Orwell) |
| Int | RemaindUserAfterMinutes | Amount of minutes after that the user shall get a reminder. |
| Int | TimeRecordingPrecision | Recording precision in minutes |
| Double | MaxTimeOfInactivity | Gets max. Time of inactivity before an automatic break will be started. |
| Guid | DefaultProjectId | Default project id to start the first logging |
| String | DefaultProjectName | Default project name |
| Guid | DefaultTaskId | Default task id to start the first logging |
| String | DefaultTaskName | Default task name |
| Default-ProjectMode | DefaultProjectMode | Defines if the recording shall start with a fixed project or with the project that has been used for the last time recording. |
| UserProfileDto | UserProfile | The user profile to which the logging settings belongs. |

TimePunch Mailing Service

The TimePunch contains all DTOs needed to send emails mailing service. These include in particular the addresses of DTOs.

PlainMailAddressDto

This class is used for mail recipients that are not associated with any user profile in TimePunch.

| Datatype | Name | Description |
|------------------------|-----------------|---|
| MailAddressType | MailAddressType | Defines the address type (From, Sender, To, CC, Bcc). |
| String | MailAddress | Mail Address |
| String | DisplayName | Display name of the mail receiver/sender |

UserMailAddressDto

This class is used for all mail recipients that can be mapped to a user profile in TimePunch.

| Datatype | Name | Description |
|------------------------|-----------------|---|
| MailAddressType | MailAddressType | Defines the address type (From, Sender, To, CC, Bcc). |
| Guid | UserId | Id of the TimePunch User Profile. |

MailAttachementDto

This class contains all the data of an attachment that can be sent by the TimePunch mailing service.

| Datatype | Name | Description |
|---------------|----------|-------------------------------------|
| String | FileName | File name of the attachement. |
| Byte[] | Content | Content of the attachement to send. |
| String | MimeType | Mime Type of the attachement |

TimePunch Project Service

ProjectDto

This class contains all data of a project.

| Datatype | Name | Description |
|---------------|-------------------------|--|
| Guid | Id | Unique ID of the project |
| DateTime | LastUpdate | Time stamp of the last update |
| Bool | IsPaid | Indicates that the project will be paid If true, PricePerHour, Vat and Currency must be filled. |
| Double | PricePerHour | Defines the hourly rate |
| Double | Vat | Defines the VAT |
| String | Currency | Defines the used currency. E.g. € |
| Long | TextColor | Text colour of the project. |
| Long | BackgroundColor | Background colour of the project |
| String | ProjectName | Name of the project |
| String | ProjectDescription | Short description of the project |
| Bool | IsProjectDeleted | True, if the project has been deleted |
| Bool | IsProjectTimeLimited | True, if the project is limited in time. If true, StartDate and EndDate must be filled |
| DateTime? | StartDate | The first valid booking date for the project, or NULL if the project is not time limited. |
| DateTime? | EndDate | The last valid booking date for the project, or NULL if the project is not time limited. |
| List<TaskDto> | TaskList | Collection of tasks that belongs to the project |
| Bool | IsWorkingTimeRestricted | True, if the project has a maximum amount of working hours. If true, MaxWorkingTime must be filled |
| Datatype | Name | Description |
| Double | MaxWorkingTime | Maximum Amount of Working Time |
| CustomerDto | Customer | Default customer that is used when creating new time entries. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|-------------------------|-------------------------|---|
| ProjectState | ManualProjectState | Current project state. Calculated = 0, Green = 1, Yellow = 2, Red = 3 |
| String | ProjectStateDescription | User defined description of the project state |
| Bool | DontInheritCommonTasks | True, if the client shall not show the common tasks at the project level. |
| Bool | IsOrderBased | True, if the project is based on a customer order |
| String | Account | Project account number |
| String | CostCenter | Cost center of the project |
| OrderType | OrderType | The order type of the project Undefined = 0 PrivateSector = 1 PublicService = 2 SupplyOfTemporaryWorkers = 3 ServiceContract = 4 ContractToProduceAWork = 5 |
| String | ContractId | Order nr. of the project |
| String | PlaceOfServiceProvision | The place of service provision |
| Double? | ChargingJourneyTime | Percentage of driving time hat counts as working time |
| Int | RoundPrecisionInMinutes | Rounding precision in minutes |
| Rounding | Rounding | Type of rounding NotRounded = 0 Rounded = 1 RoundUp = 2 RoundDown = 3 |
| Double | YellowTrafficLight | Factor where the traffic light turn to yellow |
| Double | RedTrafficLight | Factor where the traffic light turn to red |
| ProjectLifeCycle | ProjectLifeCycle | Current life cycle state of the project |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------------------------|----------------------------|---|
| ProjectAvailability | ProjectAvailability | Defines from whom the project can be accessed. AvailableForAll = 0 AvailableForSelected = 1 AvailableForNone = 2 |
| UserProfileDto | ProjectOwner | Defines the project owner |
| Guid? | ParentProjectId | Id of the parent project or NULL, if there is no parent project defined. |
| String | ParentProjectName | Read only name of the parent project |
| String | ParentProjectDescription | Read only description of the parent project |
| String | FurtherAccountingDetails | User specific data used to concrete the project accounting |
| String | ProjectCode | The barcode of a project. Used to identify a project by a barcode scanner. |
| Bool | IsMaxWorkingTimeExceedable | True, if the maximum working time of a project can be exceeded |

ProjectSearchDto

This class is derived from the PagingContextDto and contains a filter in order to search for projects.

| Datatype | Name | Description |
|-------------------------|-------------------------------|--|
| String | ProjectName | Name of the project to search |
| Bool | InTimeFrame | If true, only projects that are active and in the bookable time frame will be returned |
| Bool | EnrichWithTasks | If true, the projects will be loaded with included task information. |
| String | GenericSearch | Generic search will search for project name and description |
| ProjectLifeCycle | ProjectLifeCycle | Searches only for projects that matches the defined lifecycle |
| Guid? | ParentProjectId | If set, returns only the projects of the given parent |
| Bool | EnrichWithUserDefinedCustomer | If true, the projects will be enriched with customer data |

TIMEPUNCH API SPECIFICATION 3.4

ProjectTaskDto : TaskDto

This class is derived from TaskDto and contains the name of the project to which the task belongs as an additional property.

| Datatype | Name | Description |
|----------|---------|---|
| String | Project | Name of the project to which the task belongs |

ProjectTimeDto

This class is used to retrieve the time budget of a project.

| Datatype | Name | Description |
|-----------|---------------|--|
| DateTime? | Date | NULL, if all project times shall be looked up. A special date if only the times up to the passed date shall be calculated. |
| Guid | UserId | Id of the TimePunch User that has been looked up. |
| Guid | ProjectId | Id of the project that has been looked up. |
| Double | ScheduledWork | The scheduled work for the project. |
| Double | HoursWorked | The hours the user has already been worked on the project. |
| Double | AvailableWork | The available work that can be booked by the user. |

ProjectUserDto : UserProfileDto

This class contains the combination of Project and Profile information. It will be used read only to identify which profiles can book times to the distinct projects. The class is derived from the UserProfileDto.

| Datatype | Name | Description |
|------------|---------------|---|
| Guid | Id | Unique ID of TimePunch Profile |
| String | ProjectName | Name of the project |
| Guid | ProjectId | Id of the project |
| Double | ScheduledWork | The scheduled work for the user. |
| PoolType | PoolType | The pool type (allowed values are Shared / Private) |
| UserRights | UserRights | Contains the permissions of the user that belongs to the project. This field is used to identify project leaders. |

TIMEPUNCH API SPECIFICATION 3.4

ProjectUserRelationSaveDto : UserProfileDto

This class contains user-specific settings for a project. This class is derived from UserProfileDto.

| Datatype | Name | Description |
|----------|---------------------|--|
| Guid | ProjectId | Id of the project |
| Guid? | CustomerId | Id of the default customer if set, or NULL if no default customer has been set. |
| Double? | PricePerHour | Price per hour if set, or NULL if no user specific price per hour has been set. |
| Double? | MaxWorkingTime | Maximum working time for the user in the specified project. Or NULL, if no maximum working time for the user has been set. |
| Double? | ChargingJourneyTime | Percent of charging journey time for the user in the specified project. Or NULL, if no charging journey time has been set. |

ProjectUserRelationDto : ProjectUserRelationSaveDto

This method is derived from *ProjectUserRelationSaveDto* and contains additional information that is only returned when loading the data.

| Datatype | Name | Description |
|------------|---------------|-----------------------------------|
| String | Project | Name of the project |
| String | CustomerRefNr | Reference number of the customer |
| String | CustomerName | Name of the customer |
| UserRights | UserRights | The permission of the given user. |

TaskDto

This class contains all data that describes a task. A task can be attached directly to a project, or standalone as a common task that is not bound to a project. Common Tasks can be used in combination with any project.

| Datatype | Name | Description |
|---------------|-----------------|---|
| Guid | Id | Unique ID of the task |
| String | TaskNameId | Name of the task |
| String | TaskDescription | Description of the task |
| Bool | IsDeleted | True, if the task is deleted and cannot be used for new time entries. |
| TimeEntryType | Usage | Defines how the task entry will be booked when creating new time entries. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|---------------|------------|--|
| Double | CostFactor | The cost factor defines a factor that multiplied by the hourly rate, defines the net cost. |
| String | TaskCode | Barcode of the task. Used to identify a task by the barcode scanner. |

TimePunch Reporting Service

The reporting service contains many DTOs which can be used for the preparation of reports.

ReportCustomerDto

This class contains all customer data that is relevant for report generation.

| Datatype | Name | Description |
|--------------------------------|------------------------------|--|
| Guid | CustomerId | Unique Id of the Customer in TimePunch |
| String | CustomerName | Name of the customer |
| String | CustomerRefNr | Ref. No. of the customer |
| String | Website | URL of the customers website |
| String | ContactPerson | Contact person of the customer |
| String | ContactPersonJobPosition | Job that the contact person of the customer owns. |
| String | EMailAddress | Email Address of the customer |
| String | PostalAddress | Postal address of the customer |
| String | PhoneNumber | Phone number of the customer |
| String | FaxNumber | Fax number of the customer |
| String | MobileNumber | Mobile number of the customer |
| String | Notes | Notes of the customer |
| String | VatRegNo | Vat No. of the customer |
| ColumnFlagsFor TimeSheetExport | ColumnFlags | [Currently not used] Columns for time sheet export to the customer |
| String | ExportTimeSheet MailTemplate | [Currently not used] Mail template for the time sheet export |
| SyncTimeSheetFlags | SyncSheetFlags | [Currently not used] Timesheet synchronize options |
| Bool | IsCustomerDeleted | True, if the customer has been deleted in TimePunch. |
| String | CustomerCode | Barcode of a customer. Used to identify the customer by its barcode. |

TIMEPUNCH API SPECIFICATION 3.4

ReportDailySummaryDto

This class contains the report data for daily reporting.

| Datatype | Name | Description |
|------------------|-------------------------|--|
| ReportSummaryDto | Summary | Reference to the monthly summary |
| ReportWorkdayDto | Workday | Workday model for the reported working day. |
| DateTime | Date | Reported date |
| DateTime? | LogonTime | Earliest logon at the date |
| DateTime? | LogoffTime | Latest logoff at the date |
| String | Description | Description text of the reported date |
| Double | AbsDailyOvertime | Absolute daily overtime. This value is always >0 or 0. |
| Double | AbsDailyMinustime | Absolute daily minus time. This value is always >0 or 0. |
| Double | DailyEstimatedWorktime | Estimated working time at the given date. |
| Double | DailyEstimatedPaidtime | Estimated paid time at the given date. |
| Double | DailyOvertime | Daily overtime, this value can also be negative. |
| Double | DailyBankHoliday | Daily bank holiday hours |
| Double | DailyBankHolidayAsDays | Daily bank holiday as part of day |
| Double | DailySpecialLeave | Daily special leave hours |
| Double | DailySpecialLeaveAsDays | Daily special leave as part of day |
| Double | DailyAbsence | Daily absence hours |
| Double | DailyAbsenceAsDays | Daily absence as part of day |
| Double | CurrentOvertime | Current overtime at the given date |
| Double | DailySickComplete | Daily sick and sick not paid hours |
| Double | DailySickCompleteAsDays | Daily sick and sick not paid as part of day |
| Double | DailySickNotPaid | Daily sick not paid hours |
| Double | DailySickNotPaidAsDays | Daily sick not paid as part of day |
| Double | DailySick | Daily sick hours |
| Double | DailySickAsDays | Daily sick as part of day |
| Double | DailyTakenOvertime | Daily taken overtime |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|---------------|-----------------------------|---|
| Double | DailyTakenOvertimeAsDays | Daily taken overtime as part of day |
| Double | DailyLeave | Daily leave hours |
| Double | DailyPaidLeave | Daily paid leave hours, This can vary from the daily leave if the payout gets calculated by the last 13 weeks |
| Double | DailyLeaveAsDays | Daily leave as part of day |
| Double | DailySpareTime | Daily spare time hours |
| Double | DailySpareTimeAsDays | Daily spare time as part of day |
| Double | DailyBreaktime | Daily break time hours |
| TimeEntryType | DailyUsage | Aggregated main usage of the day |
| DateTime? | LogonTimeUncut | Uncutted logon time, even if the cutter did run |
| DateTime? | LogoffTimeUncut | Uncutted logoff time, even if the cutter did run |
| Double | DailyNightSurchargeCoretime | Daily night surcharged core time |
| Double | DailyNightSurchargeTime | Daily night surcharged time |
| Double | DailySaturdayWorktime | Daily saturday worktime |
| Double | DailySundayWorktime | Daily Sunday worktime |
| Double | DailyPublicHolidayWorktime | Daily public holiday worktime |

ReportDSGVOInfoDto

This class contains the report data for the information form of the General Data Protection Regulation which comes into force on 25.05.2018.

| Datatype | Name | Description |
|---------------|-----------------|---|
| ReportUserDto | User | Defines the user for that the report contains data. |
| DateTime? | FirstEntry | Date/Time of the first timeentry of the user |
| DateTime? | LastEntry | Date/Time of the last timeentry of the user |
| DateTime? | LockedUpTo | Date up to when the timeentries has been locked. |
| Double | WorkdaysPerWeek | Count the workdays per week |
| Double | HoursPerWeek | Amount of working hours per week |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------------------------|--------------|--|
| List<ReportLeadingUserDto> | LeadingUsers | List of the users who have access to the personal data of the user |

ReportLeadingUserDto

This class contains all information about the users who have access to other employees.

| Datatype | Name | Description |
|------------|------------------|--|
| Guid | UserId | Defines the id of the leading user |
| String | LogonName | Logon name of the leading user |
| String | UserName | User name of the leading user |
| String | Email | Email of the leading user |
| UserRights | Rights | User rights of the leading user |
| Guid | CoveredUserId | User Id of the user that can be accessed by the leading user |
| String | CoveredLogonName | Logon name of the user that is covered by the leading user |
| String | CoveredUserName | User name of the user that is covered by the leading user |
| String | CoveredEmail | Email of the user that is covered by the leading user |
| Guid | Id | Unique identifier of the object |

ReportProjectDto

This class contains all project-related data that can be used for reporting.

| Datatype | Name | Description |
|-------------------|-----------------|---|
| ReportCustomerDto | Customer | Defines the customer that can be set for project. |
| Guid? | CustomerId | The Id of the customer or NULL if no customer has been set. |
| DateTime? | ReportingDate | The date where the reporting has been requested. |
| Guid | ProjectId | The id of the project. |
| Guid | ParentProjectId | The id of the parent project. |
| Guid | ProjectOwnerId | The Id of the project owner. The id is always an employee id. |
| String | Currency | The currency that has been defined for the project. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------------------------|-------------------------|---|
| Double | ProjectPricePerHour | The price per hour defined for the given project. |
| Double | Vat | The vat tht is used to calculate the gross amount for the current project. |
| Long | TextColor | The text color that is used to display the time entries that are related to the project. |
| Long | BackgroundColor | The background color that is used to display the time entries that are related to the project. |
| String | ProjectName | The name of the project |
| String | ProjectDescription | The description of the project |
| ProjectLifeCycle | ProjectLifeCycle | The current project life cycle of the project (e.g. Planning/Active/Done/Deleted) |
| Bool | IsProjectTimeLimited | True, if the project has a fixed ending. |
| DateTime | StartDate | The first date from that the project can be booked |
| DateTime | EndDate | The last date until that the project can be booked. |
| List<ReportTaskDto> | TaskList | Collection of project related tasks |
| ProjectAvailability | ProjectAvailability | Defines if the project is useable by all, none or only a few AvailableForAll = 0 AvailableForSelected = 1 AvailableForNone = 2 |
| List<ReportProjectUserDto> | UserRelations | Collection of users that are related to the project |
| Double? | ProjectMaxWorkingTime | The maximum amount of working time that has been defined at the project. |
| Int | RoundPrecisionInMinutes | The rounding in minutes that has been defined in the project data. |
| Rounding | Rounding | Defined rounding for the project invoicing. NotRounded = 0 Rounded = 1 RoundUp = 2 RoundDown = 3 |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|--------------|----------------------------|---|
| Double | YellowTrafficLight | The percentage of worktime to available work where the project will turn yellow |
| Double | RedTrafficLight | The percentage of worktime to available work where the project will turn red |
| Double? | ProjectChargingJourneyTime | The percentage of driving time that is calculated as working time. |
| ProjectState | ManualProjectState | The manual project state Calculated = 0 Green = 1 Yellow = 2 Red = 3 |
| String | ProjectStateDescription | The description that explains the current project state, if the project state has been set manually |
| Bool | DontInheritCommonTasks | True, if the project does not inherit common tasks, e.g. does not show it. |
| Bool | IsOrderBased | True, if the project is order based and offers special input fields for that. |
| String | Account | The internal account that can be used for worktime accumulation. |
| String | CostCenter | The cost center for the project that can be used for export tasks. |
| OrderType | OrderType | A business driven order type. Undefined = 0 PrivateSector = 1 PublicService = 2 SupplyOfTemporaryWorkers = 3 ServiceContract = 4 ContractToProduceAWork = 5 |
| String | ContractId | The project based contract id |
| String | PlaceOfServiceProvision | Address where the service will be provided. |
| Bool | IsProjectTimeSet | True, if the project has a defined amount of work |
| Double | SharedPool | Defined working time for the shared pool |
| Double | PrivatePool | Defined working time for the private pools |
| Double | ScheduledWork | Scheduled work of the project. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|--------------------------|--|
| Double | HoursWorked | Hours that have already been worked at the project. |
| Double | AvailableWork | Available hours that can be booked. |
| String | FurtherAccountingDetails | User specific data used to concrete the project accounting |
| String | ProjectCode | Barcode of a project. Used to identify a project by its barcode. |

ReportProjectTimeDto

This class contains additional information about the project in connection with the employee and the planned, completed and open project budget.

| Datatype | Name | Description |
|----------------------|---------------|--|
| ReportUserDto | User | TimePunch related user data |
| ReportProjectUserDto | ProjectUser | Project related user data |
| ReportProjectDto | Project | Project data |
| Double | ScheduledWork | Scheduled work for the user and project |
| String | PoolType | Type of the project pool S → Shared pool P → Private pool |
| Date? | Date | Reference data for that the evaluated times are valid. |
| Double | HoursWorked | Already worked hours by user for the project |
| Double | AvailableWork | Available hours that can be allocated by the user for the project. |

TIMEPUNCH API SPECIFICATION 3.4

ReportProjectUserDto

This class contains specific information about an employee in connection with a project.

| Datatype | Name | Description |
|----------|-------------------------|--|
| Guid | UserProjectId | Project Id |
| Guid | ProjectUserId | User Id |
| Bool? | IsAllowedToUseProject | True, if the user is allowed to book the project |
| Double? | UserPricePerHour | Price per hours that is defined for the user within the given project. Null, if no special price has been set. |
| Double? | UserMaxWorkingTime | Max working time that is defined for the user within the given project. Null, if no max working time has been set. |
| Guid? | UserCustomerId | Customer Id that is defined for the user within the given project. Null, if no special customer id has been set. |
| Double? | UserChargingJourneyTime | Percentage of charged journey time for the user within the given project. Null, if no special charging has been set. |

ReportSummaryDto

This class contains all the important data for monthly reporting.

| Datatype | Name | Description |
|------------------------|---------------------------|--|
| ReportYearlySummaryDto | YearlySummary | Yearly summary data |
| ReportWorkModelDto | WorkModel | Active Workmodel for the given reporting date. |
| Int | Month | Number of the reported month (e.g. 1-12) |
| Double | EstimatedWorkdaysPerMonth | Amount of estimated workdays per month |
| Double | EstimatedWorktimePerMonth | Estimated worktime per month. |
| Double | EstimatedPaidtimePerMonth | Estimated paid time per month. |
| Double | Overtime | Overtime in the given month. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|--------------------|---|
| Double | CalculatedOvertime | Calculated overtime for the given month. The calculated overtime is the total overtime before the overtime cut. |
| Double | Breaktime | Amount of breaktime in month. |
| Double | Drivetime | Amount of drivetime in month. |
| Double | Worktime | Amount of worktime in month. |
| Double | TakenOvertime | Amount of taken overtime in month. |
| Double | Leave | Amount of leave in month. |
| Double | PaidLeave | Amount of paid leave hours in month. This can vary from the leave, if the paid leave gets calculated by the average of the last 13 weeks. |
| Double | LeaveAsDays | Amount of leave days in month. |
| Double | SickComplete | Amount of sick hours and sick not paid hours in month. |
| Double | SickCompleteAsDays | Amount of sick days and sick not paid days a month. |
| Double | Sick | Amount of sick time in month. |
| Double | SickAsDays | Amount of sick days in month. |
| Double | SickNotPaid | Amount of sick not paid hours in month. |
| Double | SickNotPaidAsDays | Amount of sick not paid as days in month. |
| Double | TotalOvertime | Total overtime in month. |
| Bool | Locked | True, if the month-end closing has been done. |
| DateTime | Date | The first date in month. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|-----------------|--------------------------|--|
| OvertimeCutMode | OvertimeCutMode | Defines when the overtime gets cutted, e.g on monthly basis or at all (cumulative) |
| Bool | IsPaidOvertimeCalculated | True, if the paid overtime gets automatically calculated. |
| Double | PaidOvertime | Amount of paid overtime in month. |
| Double | DecreasePaidTime | Amount of hours that will decrease the payout to the user. |
| Double | MissingHourCompensation | Amount of working time that gets added to the monthly working time. |
| Guid | RegionId | Selected region in the current month. |
| Guid | CountryId | Selected country in the current month. |
| Double | MaximumOvertime | The maximum overtime in the current month. |
| Double | OvertimeStartsAt | Default value above the additional working time gets counted as overtime. |
| WorkContract | WorkContract | The work contract of the user Employee = 0 Freelancer = 1 Wageworker = 2 Shiftworker = 3 |
| Double? | PaidTime | Amount of paid time in month |
| Double | UnpaidTime | Amount of unpaid time in month |
| Double | BankHoliday | Amount of bank holiday in the current month. |
| Double | BankHolidayAsDays | Time at bank holidays per month. e.g. 1 equals 1 day, 0.5 equals half day |
| String | CountryName | Name of the country |
| String | RegionName | Name of the region |
| Double | PreviousMonthOvertime | Amount of overtime from the previous month. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|--------------------|--|
| Double | SpecialLeave | Amount of special leave in month |
| Double | SpecialLeaveAsDays | Amount of special leave as days in month |

ReportTaskDto

This class contains all the data for an task.

| Datatype | Name | Description |
|---------------|-----------------|--|
| Guid | TaskId | Unique Id of the task |
| String | TaskNameId | Business driven Id of the task |
| String | TaskDescription | Description of the task |
| Bool | IsTaskDeleted | True, if the task has been deleted. |
| TimeEntryType | Usage | The usage or booking type of the task. |
| Double | CostFactor | The costfactor that is provided to calculate the project invoice. |
| String | TaskCode | Barcode of the task. Used to identify the task by a barcode scanner. |

ReportTimeEntryDto

This class contains all the time entry data for report creation.

| Datatype | Name | Description |
|----------------------|-----------------|---|
| ReportCustomerDto | Customer | Customer details set to the time entry |
| ReportProjectDto | Project | Project details for the time entry |
| ReportTaskDto | Task | Task details for the time entry |
| ReportProjectUserDto | ProjectUser | Project and user related data |
| ReportUserDto | User | User related data |
| ReportWorkdayDto | Workday | Information about the workday |
| Double | PaidWorkTime100 | Defines the worktime that the user gets paid |
| Double | NetAmount | Defines the net amount the user can invoice |
| Double | GrossAmount | Defines the gross amount the user can invoice |
| Double | PricePerHour | Defines the price per hour the user can invoice |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|----------------------------------|--|
| Double | CalculatedPricePerHour | Defines the calculated price per hour, which is the PricePerHour multiplied by the cost factor of the task |
| Double? | ChargingJourneyTime | The percentage of charged journey time. The result will be displayed as work time. |
| Double? | MaxWorkingTime | The maximum working time the user can book |
| Guid | CreatedBy | Id of the user who created the time entry |
| DateTime | Created | The timestamp when the time entry has been created |
| Guid | LastUpdatedBy | Id of the user who last updated the time entry |
| DateTime | LastUpdate | The timestamp when the time entry gets the latest update |
| Bool | IsBankHolidayInherited | True, if the referenced data is a public holiday |
| Bool | IsNonRegularWorkday | True, if the user does not work regulary at this day (e.g. Sunday) |
| String | BankHoliday | Name of the bank holiday, if the IsBankHolidayInherited value is true |
| Double | NightSurcharge Coretime100 | Amount of hours that is within the night surcharge core time |
| Double | NightSurcharge Time100 | Amount of hours that is within the night surcharge time |
| Double | NightSurcharge TimeCompete100 | Amount of the night surcharge core time and the night surcharge time |
| Double | Saturday Worktime100 | Amount of Saturday work time |
| Double | Sunday Worktime100 | Amount of Sunday work time |
| Double | PublicHoliday Worktime100 | Amount of public holiday work time |
| Guid | ProjectId | Id of the related project |
| Guid | TaskId | Id of the related task |
| Guid | ProjectUserId | Id of the related project user |
| DateTime | LogonTime | Start time of the time entry |
| DateTime | LogoffTime | End time of the time entry |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|---------------|---------------------|--|
| Bool | IsPaid | Flag that defines if the time entry has been paid |
| TimeEntryType | Usage | Usage or booking type of the time entry |
| Double | BreakTime100 | Amount of booked break time |
| String | Description | Description of the time entry |
| Guid? | ProjectCustomerId | Id of the cusomter defined by the project |
| Guid | TimeEntryId | Id of the time entry |
| Double | Worktime100 | Amount of worktime |
| Bool | IsImportant | Flag that defines if the time entry is important. |
| Bool | IsOnSite | Flag that defines if the time entry is onsite. |
| Bool | IsDataPreview | Flag that defines if the time entry is a data preview entry |
| Double | LeaveAsDays | Amount of leave days |
| Double | Leave100 | Amount of leave hours |
| Double | SickAsDays | Amount of sick time as days |
| Double | Sick100 | Amount of sick time as hours |
| Double | SickNotPaidAsDays | Amount of sick time not paid as days |
| Double | SickNotPaid100 | Amount of sick time not paid as hours |
| Double | DrivingTimeAsDays | Amount of driving time as days |
| Double | DrivingTime100 | Amount of driving time as hours |
| Double | TakenOvertimeAsDays | Amount of taken overtime as days |
| Double | TakenOvertime100 | Amount of taken overtime as hours |
| Int | Days | Amount of days the time entry spans |
| Bool | IsVirtual | True, if the time entry is only virtual - means calculated and not manually booked by the user |
| String | CreatedByUser | Name of the user who created the time entry |
| String | LastUpdateByUser | Name of the user who did the last update |
| Double | Duration100 | The duration of the time span |
| Double | DurationAsDay | The duration of the time span in fraction of the day |

TIMEPUNCH API SPECIFICATION 3.4

ReportUserDto

This class contains all user-specific data for reporting.

| Datatype | Name | Description |
|-------------------|---------------------------------|--|
| Guid | UserId | Unique Id of the TimePunch Profile |
| String | LogonName | Unique Logon name of the TimePunch Profile |
| String | UserName | User name of the TimePunch Profile |
| String | PersonnelNumber | Given personnel number. |
| String | UserCode | Barcode of the staff member. Used to identify a user by its barcode. |
| Double | DefaultOvertimeStartsAt | Default value above the additional working time gets counted as overtime. |
| LeaveCarryForward | DefaultLeaveCarryForward | Value that defines when the new leave shall be added to the staff member. Allowed values are BeginOfJanuary through EndOfDecember. |
| LeaveExpiration | DefaultLeaveExpiration | Defines when the leave of the previous year will expire. Allowed values are Instantly, Never and EndOfJanuary through EndOfDecember. |
| UserRights | Rights | The current user profile rights (e.g. None, View, Edit, Full, Human Resource or Admin) |
| OvertimeCutMode | DefaultOvertimeCutMode | Defines if the overtime shall be cut. Possible values are None, Monthly or Cumulative |
| Bool | DefaultIsPaidOvertimeCalculated | True, if the overtime gets automatically paid. |
| DateTime | DefaultBeginOfWork | The default time, when the user starts working. This can be overwritten by any working time model. |
| WorkContract | DefaultWorkContract | Work contract that is set for the user in the profile. |
| Double | PreviousOvertime | The overtime with that TimePunch starts the overtime calculation at beginning of the first month. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|------------------|----------------------------|--|
| Double | PreviousSickdays | The amount of sick days with that TimePunch starts the sick day counter at the beginning of the first month. |
| String | SaveAsName | The user name of the TimePunch Profile if set, or the logon name. |
| Double | DefaultChargingJourneyTime | Defines how many percentage of the driving time will be booked as working time. |
| Guid | DefaultRegionId | The default region id, if no other has been defined. |
| Guid | DefaultCountryId | The default country id, if no other has been defined. |
| Double | DefaultAnnualVacation | Default value of the annual vacation. |
| LeaveCalculation | DefaultLeaveCalculation | <p>Defines the default, if user leave is based per hours or per days.</p> <p>Days = 0 Hours = 1</p> |
| Double | DefaultLeavePerDay | Amount of hours for a leave day |
| Double? | DefaultPaidTime | The time that the user gets paid per default. |
| Double | DefaultMaximumOvertime | Default value of the maximum overtime settings for the profile. |
| LeaveSetting | LeaveSetting | <p>Defines how the leave offset shall be used for calculation.</p> <p>TakeAsOffset => In the first year, the user has extra holiday available, because he/she carry it forward from the previous time recording.</p> <p>TakeAsStatic => In the first year, the user does not own the complete holidays, but the one he entered</p> |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|------------------|---------------------------------|--|
| LeaveCalculation | DefaultPreviousLeaveCalculation | Defines the default for the previous or initial leave calculation. The previous leave can be based on hours or days. |
| Double | DefaultPreviousLeavePerDay | The amount of hours that have been previously set as default to define the length of a leave day. The value is hourly based and only used if the previous holiday is defined per hours. |
| Double? | DefaultRemainingLeave | The leave that will be used instead of the yearly holidays. Only used when LeaveSetting is set to TakeAsStatic. The value is hourly based and NULL, if the leave is counted in days. |
| Double | DefaultRemainingLeaveAsDays | The leave that will be used instead of the yearly holidays. Only used when LeaveSetting is set to TakeAsStatic. |
| Double? | DefaultAdditionalLeave | Leave that will be added to the annual leave in the first year. Only used when LeaveSetting is set to TakeAsOffset. The value is hourly based and NULL, if the leave is counted in days. |
| Double | DefaultAdditionalLeaveAsDays | Leave that will be added to the annual leave in the first year. Only used when LeaveSetting is set to TakeAsOffset. |
| DateTime | LastSyncTime | Timestamp when the user has been synced time entries. |
| UserAccountType | UserAccountType | Type of the TimePunch Profile Account (UserAccount, TechnicalAccount) |
| String | Email | Email Address of the user |
| Bool | IsUserInCoverage | True, if the user gets covered by the Authenticated Identity Profile. |
| String | TimeZone | Time Zone Information for the user. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|-----------|--------------------------------|---|
| String | Team | Information about the team to that the user belongs |
| String | Department | Information about the department to that the user belongs |
| String | Branch | Information about the branch to that the user belongs. |
| String | Birthname | Name at birth of the user |
| String | PrivatePostalAddress | Postal address of the user |
| DateTime? | Birthday | Birthday of the user |
| String | Birthplace | Place where the user has been born |
| String | Nationality | Nationality of the user |
| DateTime? | DateOfJoining | Date where the user joined the company |
| String | InsurancePolicyNumber | Number of the insurance policy of the user |
| Bool | IsMainJob | True, if the current job is the main job of the user. |
| Bool | HasOtherJobs | True, if the user has other jobs beside the current one. |
| String | JobDescription | Description of the current job the user is employed for. |
| Bool | IsRelatedToEmployer | True, if the user has a relation to the employer (e.g. brother etc) |
| Bool | IsFreedFromPensionInsurance | True, if the user is freed from pension insurance |
| String | BankingInstitut | Name of the banking institute for the wage payment or salary |
| String | InternationalBankAccountNumber | IBAN Number to pay the wage or salary |
| String | BankIdentifierCode | Bank identifier code |
| Double | HourlyWage | Hourly wage that the user gets paid. |
| Double | Salary | Monthly salary the user gets paid. |
| DateTime? | DateOfLeaving | Date where the user left the company. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|----------------|-----------------------|
| String | Md5PassHash | MD5 Password hash |
| String | Sha256PassHash | SHA 256 Password hash |
| String | PasswordSalt | Password salt value |

ReportWorkdayDto

This class defines the working time model for a single working day.

| Datatype | Name | Description |
|----------|---------------------------|--|
| Bool | HasToWork | True, if the user has commonly to work the day. |
| Double | WorkmodelDefinedWorkTime | The amount of worktime that is defined for the day. |
| Double | WorkmodelDefinedPaidTime | The amount of paid time that is defined for the day. |
| DateTime | BeginOfWork | The planned begin of the workday |
| DateTime | EndOfWork | The planned end of the workday |
| Bool | IsBreakDefined | True, if a break has been defined for the workday. |
| Double | WorkmodelDefinedBreakTime | The breaktime that has been defined for the workday. |
| Double | ForceBreakAbove | The amount of work aboe that the break is mandatory |
| Double | EstimatedDuration | The estimated duration of the workday (public holidays will be evaluated too) |
| Double | EstimatedWorktime | The estimated worktime of the workday (public holidays will be evaluated too) |
| Double | EstimatedPaidtime | The estimated paidtime of the workday (public holidays will be evaluated too) |
| Double | EstimatedBreaktime | The estimated breaktime of the workday (public holidays will be evaluated too) |
| Double | EstimatedFractionOfDay | The estimated fraction of the workday (public holidays will be evaluated too). |

IMPORTANT: The Estimated... Properties will evaluate the public holidays, which is the main difference to the other fields.

TIMEPUNCH API SPECIFICATION 3.4

ReportWorkModelDto

This class contains all the user-specific data of a working time model for reporting.

| Datatype | Name | Description |
|----------|----------------------------|---|
| String | DisplayName | Displayname of the work model. |
| DateTime | ValidSince | Date starting from that the work model is used. |
| Double | WorkdaysPerWeek | Amount of work days per week. |
| Double | HoursPerWeek | Amount of working hours per week. |
| Bool | SundayHasToWork | True, if Sunday is a regular working day. |
| Double | SundayEstimatedWorkTime | Estimated working time at Sunday. |
| Bool | MondayHasToWork | True, if Monday is a regular working day. |
| Double | MondayEstimatedWorkTime | Estimated working time at Monday. |
| Bool | TuesdayHasToWork | True, if Tuesday is a regular working day. |
| Double | TuesdayEstimatedWorkTime | Estimated working time at Tuesday. |
| Bool | WednesdayHasToWork | True, if Wednesday is a regular working day. |
| Double | WednesdayEstimatedWorkTime | Estimated working time at Wednesday. |
| Bool | ThursdayHasToWork | True, if Thursday is a regular working day. |
| Double | ThursdayEstimatedWorkTime | Estimated working time at Thursday. |
| Bool | FridayHasToWork | True, if Friday is a regular working day. |
| Double | FridayEstimatedWorkTime | Estimated working time at Friday. |
| Bool | SaturdayHasToWork | True, if Saturday is a regular working day. |
| Double | SaturdayEstimatedWorkTime | Estimated working time at Saturday. |

TIMEPUNCH API SPECIFICATION 3.4

ReportYearlySummaryDto

This class contains all the data that is important for annual leave and reporting.

| Datatype | Name | Description |
|-------------------|----------------------------------|---|
| Int | Year | Current year. E.g 2015 |
| LeaveCarryForward | LeaveCarryForward | Defines the month, when the remaining leave will be carried forward to the next year. |
| LeaveExpiration | LeaveExpiration | Defines the month, when the previous leave will expire in the current calculation period. |
| DateTime | StartOfYearlySummary | Date when the calculation of the new year starts. |
| DateTime | EndOfYearlySummary | Date when the calculation of the year ends |
| DateTime | LeaveExpirationDate | Date when the previous leave expires. |
| Double? | AdditionalLeave | Amount of additional leave hours from the previous year. |
| Double | AdditionalLeaveAsDays | Amount of additional leave days from the previous year. |
| LeaveCalculation | LeaveCalculation | Defines if the leave calculation is based on days or hours. |
| Double? | AnnualLeave | Amount of annual leave hours. |
| Double | AnnualLeaveAsDays | Amount of annual leave days |
| Double | LeavePerDay | Amount of hours that are calculated for one leave day. |
| Double | LeaveInYear | Taken leave hours in the current year. |
| Double | LeaveAsDaysInYear | Taken leave days in the current year. |
| Double | LeaveInYearUntilExpiration | Taken leave hours until the old leave expires. |
| Double | LeaveAsDaysInYearUntilExpiration | Taken leave days until the old leave expires. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|-------------------------------------|--|
| Double | LeaveInYearNewPeriod | Taken leave hours in the new leave period of the current year. |
| Double | LeaveAsDaysInYearNewPeriod | Taken leave days in the new leave period of the current year. |
| Double | SickAsDaysInYear | Amount of sick days in year |
| Double | SickNotPaidAsDaysInYear | Amount of sick days that are not paid in the current year. |
| Double | SickCompleteAsDaysInYear | Amount of sick days and sick days not paid in year. |
| Double? | RemainingLeaveUntilExpiration | Remaining leave as hours until the old leave expires. NULL, if the leave is counted in days. |
| Double | RemainingLeaveAsDaysUntilExpiration | Remaining leave as days until the old leave expires. |
| Double? | RemainingLeaveNewPeriod | Remaining leave as hours in the new period. NULL, if the leave is counted in days. |
| Double | RemainingLeaveAsDaysNewPeriod | Remaining leave as days in the new period. |

TimePunch Summary Service

DailySummaryDto

This entity contains the summary of the working time per day.

| Datatype | Name | Description |
|----------------|---------------------|--|
| UserProfileDto | User | TimePunch Profile for that the entity contains the daily summary. |
| DateTime | Date | Date for that the daily summary has been retrieved. |
| DateTime? | LogonTime | Earliest start time |
| DateTime? | LogoffTime | Latest end time |
| String | Description | Manual descriptions |
| Double | EstimatedWorktime | Amount of the estimated working time |
| Double | EstimatedPaidtime | Amount of the estimated paid time (this can vary because of the leave payment) |
| Double | Worktime | Real logged working time |
| Double | Overtime | Overtime |
| Double | Sick | Sick time |
| Double | SickAsDays | Sick time per day. e.g. 1 equals 1 day, 0.5 equals half day |
| Double | TakenOvertime | Taken overtime |
| Double | TakenOvertimeAsDays | Taken overtime per day. e.g. 1 equals 1 day, 0.5 equals half day |
| Double | PaidLeave | Amount of leave that gets paid to the user |
| Double | Leave | Time booked as leave |
| Double | LeaveAsDays | Time booked as leave. e.g. 1 equals 1 day, 0.5 equals half day |
| Double | Breaktime | Break between working times |
| Double | BankHoliday | Time at bank holidays that are relevant for payment on public holidays. |
| Double | BankHolidayAsDays | Time at bank holidays per day. e.g. 1 equals 1 day, 0.5 equals half day |
| Double | SpecialLeave | Time that is booked as a special leave. |
| Double | SpecialLeaveAsDays | Time booked as special leave per day. |
| TimeEntryType | Usage | The main time entry usage of the day. Valid values are Worktime, DrivingTime, Leave, Sickness, TakeOvertime, |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|------|---|
| | | SpecialLeave, OfficialBankHoliday, Weekend, Weekday, MissingWorkday |

MonthlySummarySaveDto

Using this DTO the conditions for the monthly working time calculation can be set.

| Datatype | Name | Description |
|-----------------|--------------------------|---|
| Guid | Id | Unique ID of the monthly summary |
| DateTime | Date | Date of the month that shall be saved |
| WorkContract | WorkContract | Type of work contract for the user. Employee = 0 Freelancer = 1 Wageworker = 2 Shiftworker = 3 |
| Double? | PaidTime | The amount of time that the user gets paid, NULL if the amount gets calculated |
| CountryDto | Country | The country information used for the holiday calculation. |
| RegionDto | Region | The region information used for the holiday calculation. |
| OvertimeCutMode | OvertimeCutMode | The cut mode defines if the overtime shall be cut monthly, cumulative or even not cut. |
| Double | OvertimeStartsAt | Amount of additional working time that is included in the working contract. Starting at this number, the working time is called overtime. |
| Double | MaximumOvertime | The maximum amount of overtime that a user can have in the OvertimeCutMode range. |
| Bool | IsOvertimeAboveLimitPaid | True, if the amount of overtime that exceeds the maximum overtime shall be paid out. |
| Double | PaidOvertime | Paid out overtime. Either manually input or calculated. |
| Double | DecreasePaidTime | The value will decrease the payout |
| Double | MissingHourCompensation | Used to compensate missing overtime. |

TIMEPUNCH API SPECIFICATION 3.4

MonthlySummaryDto : MonthlySummarySaveDto

This class contains the summary of the working times for one month.

| Datatype | Name | Description |
|----------------|--------------------|--|
| UserProfileDto | User | User profile for that the monthly summary has been created. |
| Bool | Locked | True, if the current month has been locked. |
| Double | EstimatedWorktime | Defines the estimated working time for the user. |
| Double | EstimatedPaidtime | Defines the estimated paid time for the user. |
| Double | Worktime | Real logged working time |
| Double? | PreviousOvertime | Overtime of the previous month. |
| Double? | Overtime | Overtime of the current month. |
| Double? | CalculatedOvertime | Gets the originally calculated overtime |
| Double? | TakenOvertime | Taken overtime of the current month |
| Double? | TotalOvertime | Total overtime at the end of the month. |
| Double | Breaktime | Amount of breaktime in the month. |
| Double | Drivetime | Amount of driving time in the month. |
| Double | Sparetime | Amount of spare time (it's the time that the user has been taken off - regardless what reason) in the month. |
| Double | Leave | Amount of leave in the current month. |
| Double | LeaveAsDays | Amount of days that the user has been booked as Leave. |
| Double | PaidLeave | Amount of leave that gets paid, this can vary from the leave the user has taken. |
| Double | SickComplete | Amount of sick time and sick not paid hours |
| Double | SickCompleteAsDays | Amount of sick time and sick not paid days |
| Double | Sick | Amount of sick in the current month. |
| Double | SickAsDays | Amount of days that the user has been booked as Sickness. |
| Double | SickNotPaid | Amount of sick time that is not paid |
| Double | SickNotPaidAsDays | Amount of sick days that are not paid |
| Double | BankHoliday | Time at bank holidays that are relevant for payment on public holidays. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|----------|--------------------|--|
| Double | BankHolidayAsDays | Time at bank holidays per day. e.g. 1 equals 1 day, 0.5 equals half day |
| Double | SpecialLeave | Amount of special leave the user has taken |
| Double | SpecialLeaveAsDays | Amount of special leave the user has taken in days |
| Double | UnpaidTime | The unpaid time is the difference of the paid time to the regular working time |

SummaryLockDto

This class contains the data for the month-end closing.

| Datatype | Name | Description |
|----------------|------------|---|
| UserProfileDto | User | TimePunch User Profile |
| DateTime? | LockedUpTo | Date until that the user summary has been closed. |

TimeAccountDto

This class contains the summary of the working time data for a special TimePunch Profile for the current month.

| Datatype | Name | Description |
|----------------|--------------------------|--|
| UserProfileDto | User | TimePunch User Profile |
| Double | WorktimeInMonth | Work time in the given month |
| Double | SickDaysInYear | Amount of sick days in the given year |
| Double? | TotalOvertimeUntilToday | Amount of overtime in sum until the current day |
| Double | PlannedLeaveDaysInYear | Planned leave days in the current year |
| Double? | UnplannedLeaveDaysInYear | Amount of holidays the user did not planned, within in the current year. |
| Double? | AnnualLeave | Amount of holidays in the current year. |
| TimeCutAnalyse | TimeCutAnalyse | Defines if a time cut did take place, or if it is foreseeable. NoCut = 0, Foreseeable = 1, Cutted = 2 |
| WorkContract | WorkContract | Workcontract of the user |

TIMEPUNCH API SPECIFICATION 3.4

YearlySummarySaveDto

This DTO is used to amend the yearly holiday calculation.

| Datatype | Name | Description |
|-------------------|-------------------|---|
| Guid | Id | Unique ID of the yearly summary |
| Int | Year | The year for that the summary is valid |
| LeaveCarryForward | LeaveCarryForward | Defines the month, when the remaining leave will be carried forward to the next year. |
| LeaveExpiration | LeaveExpiration | Defines the month, when the previous leave will expire in the current calculation period. |
| Double | AnnualLeaveAsDays | The amount of leave days in the given year. |
| Double? | AnnualLeave | The amount of leave hours in the given year, NULL if the leave gets calculated in days. |
| Double | LeavePerDay | Defines the amount of hours that will be calculated for a taken leave day |
| LeaveCalculation | LeaveCalculation | Defines if the leave will be calculated in days or hours Days = 0 Hours = 1 |

YearlySummaryDto : YearlySummarySaveDto

This DTO contains all information to display the yearly holiday account of a profile. The class is derived from the YearlySaveSummaryDto

| Datatype | Name | Description |
|----------------|--------------------------------|---|
| UserProfileDto | User | The user for which the summary data is loaded. |
| DateTime | StartDate | The first date in the calculation period. |
| DateTime | LastDate | The last date in the calculation period. |
| Bool | Locked | True, if the yearly summary has been locked. That is the case if one month in the calculation period has been locked. So it will be set indirectly. |
| DateTime | LeaveExpirationDate | Defines the date when the previous leave will expire. |
| Double | LeaveInYear UntilExpiration | Amount of leave hours in the current year until the expiration date. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|------------------|---|---|
| Double | LeaveAsDaysInYear UntilExpiration | Amount of leave days in the current year until the expiration date. |
| Double | LeaveInYear NewPeriod | Amount of leave hours in the current year after the expiration date. |
| Double | LeaveAsDaysInYear NewPeriod | Amount of leave days in the current year after the expiration date. |
| Double | LeaveInYear | Amount of leave hours in the current calculation period. |
| Double | LeaveAsDaysInYear | Amount of leave days in the current calculation period. |
| Double | SickCompleteAsDaysInYear | Amount of sick days in year and sick days not paid in year |
| Double | SickNotPaidAsDaysInYear | Amount of days marked as sickness not paid in the current calculation period. |
| Double | SickAsDaysInYear | Amount of days marked as sickness in the current calculation period. |
| Double? | RemainingLeave UntilExpiration | Amount of remaining leave hours until the expiration date. |
| Double | RemainingLeaveAsDays UntilExpiration | Amount of remaining leave days until the expiration date. |
| Double? | RemainingLeave NewPriod | Amount of remaining leave hours after the expiration date. |
| Double | RemainingLeaveAsDays NewPeriod | Amount of remaining leave days after the expiration date. |
| Double? | AdditionalLeave | The additional leave hours of the previous year that will be added to the current annual leave hours. |
| Double | AdditionalLeaveAsDays | The additional leave days of the previous year that will be added to the current annual leave days. |
| LeaveCalculation | PreviousLeaveCalculation | The leave calculation of the previous year. |
| Double | PreviousLeavePerDay | The previous leave hours that defined the length of a leave day. |

TimePunch Sync Service

SyncUserProfileDto

This class contains information about the TimePunch Profile and the last synchronization date.

| Datatype | Name | Description |
|-----------------|--------------|---|
| Guid | Id | Unique ID of TimePunch Profile |
| String | SaveAsName | Name of the TimePunch Profile used to show the user. |
| String | LogonName | By default the logon name is identical to the used windows logon. |
| String | UserName | First name and last name of the user. |
| DateTime | LastSyncTime | Filled, if the profile has been synced at some date. |

TimePunch TimeEntry Service

TimeEntryDto : TimeEntrySaveDto

This class contains all data of a time-entry.

| Datatype | Name | Description |
|----------------|--------------------|---|
| String | ProjectName | Name of the project that is used by the time entry |
| String | TaskName | Name of the task that is used by the time entry. |
| Double | WorkTime | Amount of work time in hours |
| Double | Duration | Complete duration of the time entry |
| Double | Leave | Amount of leave in hours |
| Double | Sick | Amount of sick time in hours |
| Double | DrivingTime | Amount of driving time in hours |
| Double | TakenOvertime | Amount of taken overtime in hours |
| Long | BackgroundColor | Background colour of the project that gets tracked. |
| Long | TextColor | Text colour of the project that gets tracked. |
| Guid | UserId | Id of the user dependent TimePunch Profile |
| String | CustomerName | Name of the customer that has been set for the time entry. |
| String | CustomerRefNr | Ref.Nr. of the customer that has been set for the time entry |
| Guid | CustomerId | Unique customer Id |
| Bool | IsVirtual | True, if the time entry is only a virtual entry, like weekend, missing workday etc. |
| Double | DurationAsDay | Duration of the time entry in relation to the workday |
| String | ProjectDescription | Description of the project referenced by the time entry |
| String | TaskNameId | Business driven task id |
| String | TaskDescription | Description of the task referenced by the time entry |
| UserProfileDto | CreatedBy | Information about the user who created the time entry. |
| UserProfileDto | LastUpdatedBy | Information about the user who last updated the time entry. |

TIMEPUNCH API SPECIFICATION 3.4

TimeEntryFilterSearchDto

This class is used to search for time entry filters.

| Datatype | Name | Description |
|-----------------|-----------------|---|
| Confidentiality | Confidentiality | Searches for time entry filters with the following confidentiality. Undefined = 0 (search private and public) Private = 1 Public = 2 |
| Bool | DeepLoad | True, if the filter shall be returned with all dependencies (projects/tasks/customers) |

TimeEntryRestrictedSaveDto

This class contains the relevant data in order to execute a restricted time entry modification. That means the user does not need elevated permissions.

| Datatype | Name | Description |
|----------|---------------|---|
| Guid | Id | Unique ID of the active time entry |
| DateTime | LastUpdate | Time stamp of the last update |
| Guid | ProjectId | Unique project Id of the time entry |
| Guid | TaskId | Unique Task Id of the time entry |
| String | Description | The description of the time entry |
| Guid | CustomerId | Unique customer Id |
| Bool | IsImportant | True, if the time entry has been marked as important. This will force the project report to show the description. |
| Bool | IsOnSite | True, if the time entry has been marked as onsite. |
| Bool | IsNotInvoiced | True, if the current entry gets not invoiced at all. |

TIMEPUNCH API SPECIFICATION 3.4

TimeEntryResultDto

This class is returned by save operations. It contains the result of the operation, as well as the data itself.

| Datatype | Name | Description |
|---------------------------------|-------------|---|
| TimeEntrySaveResult | Result | The result of the save operation. NotSaved = -1, Ok = 0, BreakEnforced = 1, EmptyBreakEntry = 2 |
| List<TimeEntryDto> | TimeEntries | List that contains the stored time entries. |

TimeEntrySaveDto

This class contains all base data in order to save a time-entry.

| Datatype | Name | Description |
|----------------------|---------------|---|
| Guid | ProjectId | Unique project Id of the time entry |
| Guid | TaskId | Unique Task Id of the time entry |
| String | LogonName | Dependent TimePunch Profile |
| DateTime | LogonTime | Logon time stamp of the time entry |
| DateTime | LogoffTime | Logoff time stamp of the time entry |
| String | Description | The description of the time entry |
| Bool | HasBeenPaid | Flag that defines if the time entry has been marked as paid. |
| TimeEntryType | Usage | Define the type of the time entry (WorkTime, DrivingTime, Leave, Sickness, TakeOvertime, BankHoliday, OfficialBankHoliday, Weekend) |
| Double | BreakTime | Amount of break time in hours |
| Guid | CustomerId | Unique customer Id |
| Bool | IsImportant | True, if the time entry has been marked as important. This will force the project report to show the description. |
| Bool | IsOnSite | True, if the time entry has been marked as onsite. |
| Bool | IsNotInvoiced | True, if the time entry gets not invoiced at all. |

TIMEPUNCH API SPECIFICATION 3.4

TimeEntrySearchDto : PagingContextDto

With this class a filter can be defined in order to search time entries. This class is derived from the PagingContextDto

| Datatype | Name | Description |
|-------------------------|--------------------|---|
| String | ReportTitle | Title of the time entry search filter |
| Confidentiality | Confidentiality | Defines if the time entry search filter is private or public - means only accessible by the person who created it, or accessible for all users. |
| DateTime | LogonTime | Search time entries starting with the given logon time |
| DateTime | LogoffTime | Search time entries through the given logoff time |
| TimeEntry-SearchPayment | Payment | Defines the type of the entries to search (SearchAllEntries, PaidEntries, NonPaidEntries) |
| List<Guid> | FilteredProjects | If null, all projects will be returned. If the collection is empty, nothing will be returned. If the collection is filled with project ids, only entries with the given project ids will be returned. |
| List<Guid> | FilteredTasks | If null, all tasks will be returned. If the collection is empty, nothing will be returned. If the collection is filled with task ids, only entries with the given task ids will be returned. |
| List<Guid> | FilteredUsers | If null or empty, the data of the current user will be returned. If the collection is filled, only entries with the given user ids will be returned. |
| Bool | ShowPublicHolidays | True, if public holidays shall be returned for the given time frame. |
| Bool | ShowWeekends | True, if the weekends shall be returned for the given time frame. |
| Bool | ShowBreaks | True, if the break times shall be shown. |
| Bool | ShowWorkingTime | True, if the working times shall be shown. |
| List<Guid> | FilteredCustomers | If null or empty, the data of the all customers will be returned. If the collection is filled, only entries with the given customer ids will be returned. |
| Bool | ShowWeekdays | True, if also weekdays shall be returned. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|------------------------------|-----------------------|--|
| Bool | ShowMissingdays | True, if also missing workdays shall be returned. Missing workdays are weekdays that are marked as working days. But for that working days are no time entry bookings available in the database. |
| DateTime? | FirstModificationTime | It defines the first modification date to search for time entries. Means all time entries that are amended after the given date. |
| DateTime? | LastModificationTime | It defines the latest modification date to search for time entries. Means all time entries that are amended before the given date. |
| Bool? | IsImportant | NULL to search both important and unimportant time entries. True to search for important and False to search for unimportant time entries |
| Bool? | IsOnSite | NULL to search both on site and offsite entries. True to search for onsite entries and false to search for offsite entries |
| Bool? | IsNotInvoiced | NULL to search both invoiced and not invoiced entries. True to search only not invoiced entries. |
| Bool | EnhanceWithAuditTrail | True, if the result shall be enhanced with information about who changed or who created the time entries. |
| TimeEntrySearch TimeFrame | TimeFrame | Used to set the logon/logoff time automatically with a constant value AllEntries = 0 UserDefined = 1 CurrentWeek = 2 CurrentMonth = 3 CurrentYear = 4 PreviousWeek = 5 PreviousMonth = 6 PreviousYear = 7 Last30Days = 8 Last3Month = 9 Last6Month = 10 Today = 11 |

TIMEPUNCH API SPECIFICATION 3.4

TimeEntrySeriesDto

Time entry series can be created using this class. Time entry series start with a specific time, but with time of day. The concrete time is taken from the work time model by TimePunch for the given user.

| Datatype | Name | Description |
|---------------|-----------------|---|
| Guid | ProjectId | Unique project Id of the time entry |
| Guid | TaskId | Unique Task Id of the time entry |
| String | LogonName | Dependent TimePunch Profile |
| DateTime | LogonDate | Logon date of the time entry |
| DateTime | LogoffDate | Logoff date of the time entry |
| TimeOfDay | LogonTimeOfDay | Time of day to start the time entry Morning = 0, Midday = 1, Evening = 2 |
| TimeOfDay | LogoffTimeOfDay | Time of day to end the time entry Morning = 0, Midday = 1, Evening = 2 |
| String | Description | The description of the time entry |
| TimeEntryType | Usage | Define the type of the time entry (WorkTime, DrivingTime, Leave, Sickness, TakeOvertime, BankHoliday, OfficialBankHoliday, Weekend) |
| Double | BreakTime | Amount of break time in hours |
| Guid | CustomerId | Unique customer Id |

TimePunch User Group Service

UserGroupDto

This class defines a group of users such as team, department or branch. Each group has group leader and zero to n assistants.

| Datatype | Name | Description |
|------------------------------|--------------|--|
| String | GroupName | Unique Name of the Group (unique in combination with the group type) |
| GroupType | GroupType | Defines the type of the group (e.g. Team, Department or Branch) |
| UserProfileDto | GroupLeader | Leader of the defined group |
| List <UserGroupMemberDto> | GroupMembers | Group Members |

UserGroupMemberDto : UserGrouSaveMemberDto

This class contains the data from loading group members. The class is derived from the UserGroupSaveMemberDto class.

| Datatype | Name | Description |
|----------|-----------|--|
| String | GroupName | Name of the group where the member belongs to. |

UserGrouSaveMemberDto

This class contains all data that is necessary to save group members. The class is derived from the UserProfileDto class.

| Datatype | Name | Description |
|-----------------|------------|---|
| Guid? | GroupId | Id of the group to that the user belongs |
| GroupType | GroupType | Type of the group to that the group belongs. |
| GroupMemberType | MemberType | Defines the type of the group member (e.g. Member or Assistant) |

UserGroupSearchDto : PagingContextDto

This class offers properties to search for user groups. This class is derived from the PagingContextDto.

| Datatype | Name | Description |
|-----------|-------------------|---|
| String | GroupName | Name of the user group to search for. |
| GroupType | GroupType | Type of the user group to search for (or undefined to search all group types) |
| Bool | EnrichWithMembers | True, if the group members shall be included within the result. |

TimePunch User Profile Service

UserProfileDataDto

This class contains all profile information that is also available at the individual methods, but combined in one export DTO. Therefore special export rights are required.

| Datatype | Name | Description |
|-------------------------|--------------------|--|
| UserProfileDto | Profile | The basic profile information (e.g. Name, email etc) |
| UserProfileDetailsDto | Details | The detailed profile information (e.g. working time contract) |
| UserProfileDefaultsDto | Defaults | The default information that is used as an offset or pre-set for internal calculations. |
| UserProfileRightsDto | Rights | The permission information for the TimePunch Profile. |
| UserProfileSensitiveDto | Sensitive | The sensitive user information for the TimePunch Profile. Those information can only be retrieved if the user has permissions to access those informations. (<i>userSensitive@export</i>) |
| Bool | IsPasswordSet | True, if the user has a password set. |
| String | PasswordWrite Only | [Write Only] Password that shall be used for the given user profile. |

UserProfileDefaultsDto : UserProfileDto

This class is derived from the UserProfileDto and contains default values that are used as the calculation base for a TimePunch profile.

| Datatype | Name | Description |
|--------------|------------------|---|
| Double | PreviousOvertime | Offset value for the overtime calculation. |
| Double | PreviousSickdays | Offset value for the sickdays. |
| LeaveSetting | LeaveSetting | Defines how the leave offset shall be used for calculation. TakeAsOffset => In the first year, the user has extra holiday available, because he/she carry it forward from the previous time recording. TakeAsStatic => In the first year, the user does not own the complete holidays, but the one he entered |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|------------------|-----------------------|---|
| Double? | AdditionalLeave | Leave hours that will be added to the annual leave in the first year. This feature is only used when LeaveSetting is set to TakeAsOffset. |
| Double | AdditionalLeaveAsDays | Leave that will be added to the annual leave in the first year. This feature is only used when LeaveSetting is set to TakeAsOffset. |
| Double? | RemainingLeave | The leave hours that will be used instead of the yearly holidays in the first year. This feature is only used when LeaveSetting is set to TakeAsStatic. |
| Double | RemainingLeaveAsDays | The leave that will be used instead of the yearly holidays in the first year. This feature is only used when LeaveSetting is set to TakeAsStatic. |
| Double | LeavePerDay | Defines the length of a leave day if leave calculation is set to hours. |
| LeaveCalculation | LeaveCalculation | Defines if the leave defaults are calculated in days or hours |
| Bool | IsLocked | [Readonly] True, if the defaults can't be changed because months are locked. |

UserProfileDetailsDto : UserProfileDto

The class is derived from UserProfileDto and contains all the detail data of the employee profile.

| Datatype | Name | Description |
|------------|---------------------|---|
| String | PersonnelNumber | Personnel number of the staff member. |
| String | UserCode | Barcode of the staff member. Used to identify the user by a barcode scanner. |
| Double | ChargingJourneyTime | The percentage of working time that shall be booked when the user enters a driving time period. |
| String | Timezone | The time zone for the user. The time zone is used to calculate the correct logon/logoff time. |
| CountryDto | Country | Country that is used for holiday calculation |
| RegionDto | Region | Region that is used for holiday calculation |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|--------------------------|--------------------------|---|
| WorkContract | WorkContract | Work Contract enumeration that specifies how the staff member shall be booked. Employee = 0 Freelancer = 1 Wageworker = 2 Shiftworker = 3 |
| Double? | PaidTime | The regular paid time, if the user is employed as a wage worker or shift worker. |
| OvertimeCutMode | OvertimeCutMode | Defines if the overtime shall be cut. Possible values are None, Monthly or Cumulative |
| Double | OvertimeStartsAt | Number of overtime that is included in the working contract. Above this limit the working time will be counted as overtime. |
| Double | MaximumOvertime | Maximum limit of overtime until which the working time will be counted as overtime. |
| Bool | IsOvertimeAboveLimitPaid | True, if the working time above the maximum overtime shall be marked as paid out. |
| LeaveCarryForward | LeaveCarryForward | Value that defines when the new leave shall be added to the staff member. Allowed values are BeginOfJanuary through BeginOfDecember. |
| LeaveExpiration | LeaveExpiration | Defines when the leave of the previous year will expire. Allowed values are Instantly, Never and EndOfJanuary through EndOfDecember. |
| Double | AnnualLeaveAsDays | Amount of leave days with that the user can plan with. |
| Double | AnnualLeave | Amount of leave hours with that the user can plan with. |
| Double | LeavePerDay | Defines the length of a leave day if leave calculation is set to hours |
| LeaveCalculation | LeaveCalculation | Defines if the leave defaults are calculated in days or hours |

TIMEPUNCH API SPECIFICATION 3.4

UserProfileImageDto : UserProfileDto

This class is used to read or update the profile images of the staff members. The class is derived from the UserProfileDto.

| Datatype | Name | Description |
|-------------|-------------|---|
| ImageFormat | ImageFormat | The used image format. For uploading only Original can be used. |
| Byte[] | ImageData | The image data as PNG or JPG Format. |

UserProfileRightsDto: UserProfileDto

This class contains the base data of the employee profile and in addition also the permission information.

| Datatype | Name | Description |
|------------|------------|---|
| UserRights | UserRights | Permission group of the user (None, View, Edit, Full, HumanResource, Admin) |

UserProfileSensitiveDto : UserProfileDto

This class contains the employee's data protection relevant information.

| Datatype | Name | Description |
|-----------|-----------------------------|---|
| string | Birthname | Birth name of the user |
| String | PrivatePostalAddress | Private postal address of the user |
| DateTime? | Birthday | Date when the user was born |
| String | Birthplace | Place where the user was born |
| String | Nationality | Nationality of the user |
| DateTime? | DateOfJoining | Date when the user joined the company |
| DateTime? | DateOfLeaving | Date when the user left the company, or NULL if the user is still employed. |
| String | InsurancePolicyNumber | The insurance policy number |
| Bool | IsMainJob | True, if the current job is the main job of the user |
| Bool | HasOtherJobs | True, if the user also have other jobs |
| String | JobDescription | Description of the work that the user does |
| Bool | IsRelatedToEmployer | Defines if the user is in any relation to his employer (e.g. sister, son etc) |
| Bool | IsFreedFromPensionInsurance | Defines if the user is freed from pension insurance. |
| String | BankingInstitut | Name of the banking institut for monthly salary submission. |

TIMEPUNCH API SPECIFICATION 3.4

| Datatype | Name | Description |
|---------------|------------------------------------|--|
| String | InternationalBank AccountNumber | IBAN Code of the employee |
| String | BankIdentifierCode | BIC Code of the banking institute |
| Double | HourlyWage | Hourly wage of the user if he/she is employed as a wage worker or shift worker |
| Double | Salary | Monthly salary of the user if he/she is employed as an employee |

TimePunch Work Model Service

WorkModelDto

This class contains the definition of one working time model.

| Datatype | Name | Required | Description |
|-------------------------------|---------------|----------|--|
| DateTime | ValidSince | Yes | The date from which of the work model is valid |
| List<WorkdayDto> | WorkdayModels | Yes | List of the workday models |

WorkdayDto

This class contains the definition for one workday.

| Datatype | Name | Required | Description |
|-----------------|----------------------|----------|--|
| DateTime | Date | Yes | Defines the date of the workday model |
| Bool | HasToWork | Yes | True, if the day is marked as a workday. |
| Double | AttendanceTime | | If it's a workday, the attendance time defines how long the staff member has to stay in the office. |
| Bool | IsTimeFrameDefined | | True, if a time frame for the workday has been defined. False, if only the working time duration has been defined. |
| DateTime | BeginOfWork | | Begin of the defined time frame. |
| DateTime | EndOfWork | | End of the defined time frame. |
| double? | BreakTime | | Not Null, if a mandatory break time has been defined. |
| Double | EstimatedWorkingTime | | The estimated working time for the given date. |
| Double | EstimatedBreakTime | | The estimated break time for the given date. |

Constant values

This chapter contains information about the constant values that are important in TimePunch.

Core

ApplicationKeys

Within the class `TimePunch.Enums.Core.ApplicationKeys` the application keys are defined. This can be used to verify the application licenses. Following methods are important for that:

- `ValidateAuthentication`
- `GetLicensedUserProfiles`

The following constant values are defined:

```
public static class ApplicationKeys
{
    /// <summary>
    /// Application Key for TimePunch PRO
    /// </summary>
    public const string TIMEPUNCH_PRO = "Tp";

    /// <summary>
    /// Application Key for TimePunch ONE
    /// </summary>
    public const string TIMEPUNCH_ONE = "One";

    /// <summary>
    /// Application Key for TimePunch TEN
    /// </summary>
    public const string TIMEPUNCH_TEN = "Ten";

    /// <summary>
    /// Application Key for PZE Terminal
    /// </summary>
    public const string PZE_TERMINAL = "PZE";

    /// <summary>
    /// Application Key for PZE Watcher
    /// </summary>
    public const string PZE_WATCHER = "WAT";

    /// <summary>
    /// Application Key for PZE Studio
    /// </summary>
    public const string PZE_STUDIO = "STU";

    /// <summary>
    /// Application Key for PZE Studio
    /// </summary>
    public const string PZE_ONLINE = "ONL";
}
```

Annex

The annex contains some helping code in order to use the TimePunch API. This isn't necessary at all, but can be very helpful.

Hash the password with the MD5 cryptography provider

The following code makes use of the MD5 cryptography provider in order to hash a password. The hashed password can be used to do the authorization with the TpAuthentication entity.

```
public static string MD5Hash(string text)
{
    System.Security.Cryptography.MD5 md5
        = new System.Security.Cryptography.MD5CryptoServiceProvider();
    return System.Text.RegularExpressions.Regex
        .Replace(BitConverter.ToString(
            md5.ComputeHash(ASCIIEncoding.Default.GetBytes(text))), "-", "");
}
```

End of the Document