



PlanningPME Web Services Documentation



Copyright © 2002-2015 TARGET SKILLS. All rights reserved.

Corporate Headquarters

TARGET SKILLS

39 Rue Michel Ange

91026 EVRY Courcouronnes Cedex

FRANCE

Web site: <http://www.planningpme.com>

Telephone: +33 (0)1 69 47 10 00

TARGET SKILLS believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS". TARGET SKILLS MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any language in any form or by any means without the written permission of TARGET SKILLS.

All trademarks herein are the sole property of their respective owners.



Summary

Technical specifications.....	4
Introduction	4
Dates format	4
Common request headers and body items	4
Common response status and headers	4
DiscoveryService.svc	6
MobileSamples method.....	6
LicensingService.svc	7
Validation method	7
DataService.svc	8
Authentication method	8
Planning method.....	9
Customers method	11
Projects method	11
Do method (create)	12
Do method (update)	12
Do method (delete)	12
File method (upload)	14
File method (update)	14
File method (delete)	15
File method (get)	15
Notification method (delete)	16
Logout method	17



Technical specifications

Introduction

PlanningPME Web Services are built on REST architecture and Microsoft WCF technologies. Data transferred are serialized in JSON.

The root URI of web services will always be referred as `http://yourdomain.com/` in the following. Replace `http://yourdomain.com/` with the root URI targeting your location.

Dates format

WCF's serializes dates in the following JSON format: `\Date(978332400000+0100)\` where 978332400000 is the number of milliseconds till Epoch, and +0100 is the time zone part.

PlanningPME Web Services deals with UTC dates only on the server-side. As a result, services will always send dates without time zone part, and the best way to format dates in a request's body is also to avoid this part (making them UTC before serialization).

Any received dates including time zone will be considered as local, into the server's time zone (request deserialization takes place on the server). That is why sending local dates can produce unattended results if server and client are not in the same time zone. Always send UTC dates to make it short.

Common request headers and body items

Some HTTP headers will always be the same or don't need to be detailed on each method, like the "User-Agent: XXX" header, which should be automatically added by your client, and the "Accept-Encoding: gzip" which is greatly recommended if you want your data to be compressed before they are transmitted in both ways.

The "Content-Type" header value should be "application/json; charset=utf-8" most of the time.

These headers won't be discussed in the following though they still need to appear in your requests.

All nullable items detailed in the JSON bodies below can be omitted, or specified as null, no matter.

Common response status and headers

Usual HTTP status is sent on each response. Status different from 200 (OK) should alert you that an error occurred during the round-trip.

The 401 status code (Unauthorized) is sent by authenticated services methods if a valid user token was not sent in the request. In such a case just call the authentication method again and repeat your request with the new user token returned.

Valid response header of almost each method in PlanningPME Web Services will be like:



Transfer-Encoding: chunked
Connection: Close
Cache-Control: private
Content-Type: application/json; charset=utf-8
Date: Wed, 21 Nov 2012 14:16:54 GMT
Server: ASP.NET Development Server/10.0.0.0
X-AspNet-Version: 4.0.30319



DiscoveryService.svc

This public service retrieves the location of each PlanningPME service.

MobileSamples method

Returns the location of the sample services dedicated to PlanningPME iPhone and Android applications.

Request	Value
URI	http://yourdomain.com/DiscoveryService.svc/MobileSamples/LANG
Method	GET
Headers	No specific header
Body	Body should be empty

- LANG = String containing the two letters ISO code of the requested language

Response	Value
Headers	No specific header
Body	[{"Lang":"LANG", "Url":"URL", "Label":"LABEL"}, ...]

- LANG = String containing the two letters ISO code of the requested language
- URL = String containing URI of the sample service
- LABEL = String containing the label of this sample



LicensingService.svc

This public service is dedicated to PlanningPME license validation.

Validation method

Validates the client's product license.

Request	Value
URI	http://yourdomain.com/LicensingService.svc/Validation
Method	POST
Headers	No specific header
Body	{"Login":"LOGIN", "Password":"PASSWORD", "Terminal":"TERMINAL", "DeviceToken":"DEVICETOKEN"}

- LOGIN = String containing the client login
- PASSWORD = String containing the client password
- TERMINAL = Nullable string containing the terminal ID of the device making the request
- DEVICETOKEN = Nullable string containing the token of the device making the request

Response	Value
Headers	No specific header
Body	{"Lock":LOCK, "Message":"MESSAGE", "ServiceUrl":"URL"}

- LOCK = Boolean containing true if this client license is locked, false otherwise
- MESSAGE = String containing the reason why license is locked
- URL = String containing URI of the corresponding data service



DataService.svc

This private service is dedicated to the reading and writing operations in a PlanningPME database.

Authentication method

Authenticates a user. Retrieves rights and a user token that should then be included in each subsequent request on this service.

Request	Value
URI	http://yourdomain.com/DataService.svc/Authentication
Method	POST
Headers	No specific header
Body	{"DeviceToken":"DEVICETOKEN", "IPad":IPAD, "Login":"LOGIN", "OS":OS, "Password":"PASSWORD", "Terminal":"TERMINAL"}

- LOGIN = String containing the user login
- PASSWORD = String containing the user password
- TERMINAL = Nullable string containing the terminal ID of the device making the request
- DEVICETOKEN = Nullable string containing the token of the device making the request
- OS = Nullable string containing one of the following:
 - « APPLE » for an Apple operating system
 - « ANDROID » for an Android operating system
 - « WINDOWS » for a Windows operating system
- IPAD = Nullable boolean containing true if the device making the request is an iPad, false otherwise

Response	Value
Headers	No specific header
Body	{"UserProfile":USERPROFILE, "UserToken":"USERTOKEN"}

- USERPROFILE = Integer containing the rights of the user, as a sum of the following enum values:
 - AddTask = 1,
 - UpdateTask = 2,
 - DeleteTask = 4,
 - ViewTask = 8,
 - AddUnavailability = 16,
 - UpdateUnavailability = 32,
 - DeleteUnavailability = 64,
 - ViewUnavailability = 128,
 - AddTaskLabel = 256
- USERTOKEN = String containing the user token that should be used to authenticate each subsequent request

An empty body is always returned without more instruction if authentication fails.



Planning method

Returns whole planning including jobs and relative items that could be used on a client application, such as tasks, customers or projects. Due to performance considerations, this method is intended to work with a client cache following three principles:

1. Each call is always limited to a given period of the planning defined by lower and upper limits relative to the moment the method is called. These two limits are set on the server, or by setting the From field within the method call (the upper limit of the period cannot be overridden during the method call). Value should be composed of a number, followed by an underscore, and a letter (M for Month, W for Week, D for Day). For instance, a value of "3_W" will get items from three weeks back, overriding the server default setting (which could be "6_M", six months).
2. Because client should not download data it already has, each response contains three parameters named LastDateSync, LastDateFrom and LastDateTo that must be saved on the client and passed to the subsequent request.
3. As PlanningPME is a multi-client application, some data of the same planning can be modified by another process, or even by another type of soft (a desktop application, an internet site etc...). That is where timestamps are useful; and there are two situations. Do objects are easily synchronized: as PlanningPME saves history of CRUD operations, server knows which object must be synchronized, added or deleted, depending on the value of LastDateSync. For associated objects (projects, customers, resources) and other useful collections (data fields, states, options, tasks and unavailabilities), no history can help and each collection will be reloaded entirely if the V parameter differs from the one calculated on the server during request. New values of V parameters like VCustomer, VProject... are included in each response, and must be saved on the client before they are passed into subsequent requests.

Clients should then know what data is obsolete in their cache and has to be deleted, added or updated.

Request	Value
URI	http://yourdomain.com/DataService.svc/Planning
Method	POST
Headers	No specific header
Body	{"UserToken":"USERTOKEN", "From":"FROM", "LastDateFrom":"LASTDATEFROM", "LastDateSync":"LASTDATESYNC", "LastDateTo":"LASTDATETO", "VCustomer":VCUSTOMER, "VDataField":VDATAFIELD, "VParameter":VPARAMETER, "VProject":VPROJECT, "VResource":VRESOURCE, "VState":VSTATE, "VTask":VTASK, "VUnavailability":VUNAVAILABILITY}

- USERTOKEN = String containing current user token
- FROM = Nullable string containing any lower limit
- LASTDATEFROM, LASTDATESYNC, LASTDATETO = String value that client has in his cache (set to null or omit if no cache)
- VCUSTOMER, VDATAFIELD, VPARAMETER, VPROJECT, VRESOURCE, VSTATE, VTASK, VUNAVAILABILITY = Integer value that client has in his cache (set to 0 or omit if no cache)



Response	Value
Headers	No specific header
Body	{ "DateFrom": "DATEFROM", "DateSync": "DATESYNC", "DateTo": "DATETO", "VCustomer": VCUSTOMER, "VDataField": VDATAFIELD, "VParameter": VPARAMETER, "VProject": VPROJECT, "VResource": VRESOURCE, "VState": VSTATE, "VTask": VTASK, "VUnavailability": VUNAVAILABILITY, "Dos": DOS, "DosDeleted": DOSDELETED, "Customers": CUSTOMERS, "DataFields": DATAFIELDS, "Parameter": PARAMETER, "Projects": PROJECTS, "Resources": RESOURCES, "States": STATES, "Tasks": TASKS, "Unavailabilities": UNAVAILABILITIES }

- DATEFROM, DATESYNC, DATETO = String value to be stored in client cache for next request
- VCUSTOMER, VDATAFIELD, VPARAMETER, VPROJECT, VRESOURCE, VSTATE, VTASK, VUNAVAILABILITY = Integer value that client should save in cache for the next request
- DOS = Array of Do objects
- DOSDELETED = Array of integer indicating which Do objects should be deleted from cache
- CUSTOMERS = Array of Customer objects *
- DATAFIELDS = Array of DataField objects *
- PARAMETER = Parameter object containing all application options *
- PROJECTS = Array of Project objects *
- RESOURCES = Array of Resource objects *
- STATES = Array of State objects *
- TASKS = Array of Task objects *
- UNAVAILABILITIES = Array of Unavailability objects *

(*) : client applications should compare V values received with values of their cache to know if the corresponding collection should be taken into account. In such a case, the collection contained in the client cache should be deleted first.



Customers method

Returns an array of Customer objects which name begins with the prefix sent.

Request	Value
URI	http://yourdomain.com/DataService.svc/Customers
Method	POST
Headers	No specific header
Body	{"UserToken":"USERTOKEN", "Prefix":"PREFIX"}

- USERTOKEN = String containing current user token
- PREFIX = String containing first letters of a customer's name. Comparison is lowercase and string should be 3 letters long at least.

Response	Value
Headers	No specific header
Body	{"Results":[CUSTOMER1, CUSTOMER2, ...]}

- CUSTOMER1 = First Customer object which name starts with the prefix sent
- CUSTOMER2 = Second Customer object...

Projects method

Returns an array of Project objects which name begins with the prefix sent.

Request	Value
URI	http://yourdomain.com/DataService.svc/Projects
Method	POST
Headers	No specific header
Body	{"UserToken":"USERTOKEN", "Prefix":"PREFIX"}

- USERTOKEN = String containing current user token
- PREFIX = String containing first letters of a project's name. Comparison is lowercase and string should be 3 letters long at least.

Response	Value
Headers	No specific header
Body	{"Results":[PROJECT1, PROJECT2, ...]}

- PROJECT1= First Project object which name starts with the prefix sent
- PROJECT2= Second Project object...



Do method (create)

Creates a new Do object. Sets Key and other calculated data (like its auto label which depends on the Planning options), then returns the new object.

Request	Value
URI	http://yourdomain.com/DataService.svc/Do
Method	PUT
Headers	UserToken: USERTOKEN
Body	DO

- USERTOKEN = String containing current user token
- DO = String containing JSON serialization of the new Do object

Response	Value
Headers	No specific header
Body	DO

- DO = String containing JSON serialization of the new Do object created

Do method (update)

Updates a Do object. Persists changes into database and returns the modified object.

Request	Value
URI	http://yourdomain.com/DataService.svc/Do/KEY
Method	PUT
Headers	UserToken: USERTOKEN
Body	DO

- KEY = Integer containing Key of the Do object to be modified
- USERTOKEN = String containing current user token
- DO = String containing JSON serialization of the Do object to be modified

Response	Value
Headers	No specific header
Body	DO

- DO = String containing JSON serialization of the persisted Do object

Do method (delete)

Deletes a Do object.



Request	Value
URI	http://yourdomain.com/DataService.svc/Do/KEY
Method	DELETE
Headers	UserToken: USERTOKEN
Body	Body should be empty

- KEY = Integer containing Key of the Do object to be deleted
- USERTOKEN = String containing current user token

Response	Value
Headers	No specific header
Body	DONE

- DONE = Boolean containing the result of operation : true if object was deleted, false if not



File method (upload)

Uploads and creates a new File object. Sets and returns Key of the new File object.

Request	Value
URI	http://yourdomain.com/DataService.svc/File
Method	PUT
Headers	Content-Type: application/octet-stream FileName : FILENAME UserToken: USERTOKEN
Body	BYTES

- FILENAME = String containing the file name with extension
- USERTOKEN = String containing current user token
- BYTES = Byte array of the file to be uploaded

Response	Value
Headers	No specific header
Body	KEY

- KEY = Integer containing Key of the new File object created

File method (update)

Uploads and updates a File object. Persists changes into database and returns the modified object.

Request	Value
URI	http://yourdomain.com/DataService.svc/File/KEY
Method	PUT
Headers	Content-Type: application/octet-stream FileName : FILENAME UserToken: USERTOKEN
Body	BYTES

- KEY = Integer containing Key of the File object to be modified
- FILENAME = String containing the file name with extension
- USERTOKEN = String containing current user token
- BYTES = Byte array of the file to be uploaded

Response	Value
Headers	No specific header
Body	KEY

- KEY = Integer containing Key of the persisted FILE object



File method (delete)

Deletes a File object.

Request	Value
URI	http://yourdomain.com/DataService.svc/File/KEY
Method	DELETE
Headers	UserToken: USERTOKEN
Body	Body should be empty

- KEY = Integer containing Key of the File object to be deleted
- USERTOKEN = String containing current user token

Response	Value
Headers	No specific header
Body	DONE

- DONE = Boolean containing the result of operation : true if object was deleted, false if not

File method (get)

Gets a File object stream.

Request	Value
URI	http://yourdomain.com/DataService.svc/File/KEY
Method	GET
Headers	UserToken: USERTOKEN
Body	Body should be empty

- KEY = Integer containing Key of the File object to get content
- USERTOKEN = String containing current user token

Response	Value
Headers	No specific header
Body	CONTENT

- CONTENT = Content of the file stored on server



Notification method (delete)

Deletes a Notification object.

Request	Value
URI	http://yourdomain.com/DataService.svc/Notification/KEY
Method	DELETE
Headers	UserToken: USERTOKEN
Body	Body should be empty

- KEY = Integer containing Key of the Notification object to be deleted
- USERTOKEN = String containing current user token

Response	Value
Headers	No specific header
Body	DONE

- DONE = Boolean containing the result of operation : true if object was deleted, false if not



Logout method

Forces the client specific data release on server.

Request	Value
URI	http://yourdomain.com/DataService.svc/Logout
Method	POST
Headers	No specific header
Body	{"UserToken":"USERTOKEN"}

- USERTOKEN = String containing current user token

Response	Value
Headers	No specific header
Body	Body should be empty

