# TFS WebService

The service is used to get the information from TFS which is not available via REST. The service is implemented as a restful service.

## **Project Structure**

The project was created using a visual studio project template and therefore contains much more classes then the really interesting ones.

### Folder “Controllers”

The “Controllers” folder contains the important Api Controller implementations. For each information type provided by the web service there is a specific controller.

HomeController (default controller)

Used for the website, not used by the service.

ProcesstemplateController

Gets the process template of a project.

SprintsController

Gets all sprints defined for a project.

TeaminfoController

Gets information about all teams of the project.

ValuesController

Tests the connection to this service.

EventController

Gets the last workitem changes of the given project since the given timestamp.

Used by the service hook of the VSO to post all events, which the web service is registered for, to the web service.

### Folder “Models”

This folder contains the data model classes used by the controller. These objects are serialized and returned by the controller as a JSON object.

* TeamInfo
* Area
* Iteration
* Sprint
* FieldChange
* WorkitemChangeEntity

### Folder “Misc”

This folder contains some extension methods used by the web service and the TFSClient class which is the one that implements all TFS API calls.

## **Api Calls**

### Get process template of the team project

Gets the process template defined for the given tfs project.

**Request**

GET  https://{webserviceaddress}/api/processtemplate?tfsuri={tfsserveraddress}&user={username}&pwd={password}&projectCollection={projectcollection}&project={projectname}

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| webserviceaddress | Url of your webservice |
| tfsserveraddress | Url of your tfs |
| username | Username for accessing the TFS |
| password | Password for the given user |
| projectcollection | The project collection to project is part of |
| projectname | Project name |

**Response**

"Microsoft Visual Studio Scrum 2013.3"

### Get sprints of a team project

Gets all sprints defined for the given project.

**Request**

GET  https://{webserviceaddress}/api/sprints?tfsuri={tfsserveraddress}&user={username}&pwd={password}&projectCollection={projectcollection}&project={projectname}

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| webserviceaddress | Url of your webservice |
| tfsserveraddress | Url of your tfs |
| username | Username for accessing the TFS |
| password | Password for the given user |
| projectcollection | The project collection to project is part of |
| projectname | Project name |

**Response**

[

{

"path":"Basta\_TFSE\\Release 1\\Sprint 2",

"startDate":"2014-11-17T00:00:00",

"endDate":"2014-11-28T00:00:00",

"name":"Sprint 2"

},

{

"path":"Basta\_TFSE\\Release 1\\Sprint 1",

"startDate":"2014-10-27T00:00:00",

"endDate":"2014-11-14T00:00:00",

"name":"Sprint 1"

},

{

"path":"Basta\_TFSE\\Release 1\\Sprint 3",

"startDate":"2014-12-01T00:00:00",

"endDate":"2014-12-12T00:00:00",

"name":"Sprint 3"

},

{

"path":"Basta\_TFSE\\Release 1\\Sprint 4",

"startDate":"2014-12-15T00:00:00",

"endDate":"2014-12-26T00:00:00",

"name":"Sprint 4"

},

{

"path":"Basta\_TFSE\\Release 1\\Sprint 5",

"startDate":"0001-01-01T00:00:00",

"endDate":"0001-01-01T00:00:00",

"name":"Sprint 5"

},

{

"path":"Basta\_TFSE\\Release 1\\Sprint 6",

"startDate":"0001-01-01T00:00:00",

"endDate":"0001-01-01T00:00:00",

"name":"Sprint 6"

}

]

### Get teams of the team project

Gets all teams of a team projects with advanced information like team name, current iteration, areas, etc.

**Request**

GET  https://{webserviceaddress}/api/teaminfo?tfsuri={tfsserveraddress}&user={username}&pwd={password}&projectCollection={projectcollection}&project={projectname}

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| webserviceaddress | Url of your webservice |
| tfsserveraddress | Url of your tfs |
| username | Username for accessing the TFS |
| password | Password for the given user |
| projectcollection | The project collection to project is part of |
| projectname | Project name |

**Response**

[

{

"TeamName":"Windows\_Team",

"TeamId":"46356f02-a5ee-4b54-9c2e-91a3276dffee",

"IsDefaultTeam":false,

"CurrentIterationInfo":"Basta\_TFSE\\Release 1\\Sprint 4",

"ActiveIterations": [

"Basta\_TFSE\\Release 1\\Sprint 2",

"Basta\_TFSE\\Release 1\\Sprint 1",

"Basta\_TFSE\\Release 1\\Sprint 3",

"Basta\_TFSE\\Release 1\\Sprint 4"

],

"Areas": [

{

"Name":"Basta\_TFSE\\Windows\_Team",

"IncludeChildren":true

}]

},

{

"TeamName":"Basta\_TFSE Team",

"TeamId":"82ab60e7-e5d2-4d6e-b775-bd0d833b4c13",

"IsDefaultTeam":true,

"CurrentIterationInfo":"Basta\_TFSE\\Release 1\\Sprint 4",

"ActiveIterations": [

"Basta\_TFSE\\Release 1\\Sprint 2",

"Basta\_TFSE\\Release 1\\Sprint 1",

"Basta\_TFSE\\Release 1\\Sprint 3",

"Basta\_TFSE\\Release 1\\Sprint 4"

],

"Areas": [

{

"Name":"Basta\_TFSE",

"IncludeChildren":true

}]

}

]

### Test connection to web service

Tests the connection to the service.

**Request**

GET  https://{webserviceaddress}/api/values

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| webserviceaddress | Url of your webservice |

**Response**

"true"

### Endpoint for the service hook

Method used by the service hook to post all events, which the web service is registered for, to the web service.

**Request**

POST  https://{webserviceaddress}/api/event

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| webserviceaddress | Url of your webservice |

### Get work item changes from Azure storage

Gets all work item changes of a project since a specific time.

**Request**

GET  https://{webserviceaddress}/api/event?projectID={projectID}&dtLastRequestTime={lastRequestDate}

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| webserviceaddress | Url of your webservice |
| projectID | The project ID |
| lastRequestDate | Timestamp since when the changes should be returned |

**Response**

TODO

## **Deploy web service**

### 3.1 Change Connection to Azure Storage

As the REST API of Visual Studio Online is not able to return a list of changed workitems for a specific time range the web service can be registered as an endpoint of a service hook of VSO. Currently only the processing of workitem change events is implemented in the service. To remember the workitem changes for e.g. refreshing of apps the web service saves the work item changes in an azure storage. To use your Azure Storage you only need to change the connection string in the web.config.

<connectionStrings>

<add name="StorageConnectionString" connectionString="DefaultEndpointsProtocol=http;AccountName=<YourAccountName>;AccountKey=<YourAccountKey>"/>

</connectionStrings>

The AccountName is the name of your azure storage. The AccountKey is one of your access keys you can create for your storage in the azure management portal.

### 3.2 Deploy the web service using Visual Studio

Open the solution in Visual Studio and right click the Project. In the context menu select “Publish” and add in the wizard your webserver informations. If you are finished press “Publish”. The project is build now and after the publish the website is opened in a browser.