# SOAP Documentation

Anthill Pro provides a SOAP-based, web services facade to enable third-party and user-created applications to cleanly integrate. The SOAP interface, available at launch, will provide a number of lookup utilities to provide information on system state, project progress, and current activity. Also, as standard build and deploy web services vocabularies evolve in the Eclipse Application Life-Cycle Framework project, this SOAP interface will provide whatever functionality is specified.

# Using the Interface

The simplest way to use the SOAP interface may be to create a Java integration using the SOAP WSDL. The WSDL provides utilities to look up the anthill3 server, and models for the data types and requests. It is the mechanism Eclipse plug-ins use to provide AnthillPro data to the Eclipse IDE. These classes are detailed in the AnthillPro SOAP WSDL. It is available at SOAP WSDL on the **Developer Tools** page.

If you would like to use SOAP, please contact [http://support.urbancode.com/] support.

For example, using Groovy you can call the client jar with something like this:

```
import com.urbancode.anthill3.integration.*

/*
** Assumes you have a standard properties file here with at least
** these 3 properties:
** user = <username>
** password = <password>
** url = <http://host:port>
*/
def filename = "${System.properties['user.home']}/.anthill/properties"

def props = new Properties()
new File(filename).withInputStream { props.load(it) }

def dashboard = new DashboardServiceLocator("${props['url']}/services/Dashboard?wsdl",
   new javax.xml.namespace.QName('integration.anthill3.urbancode.com', 'DashboardService')
   getDashboard(new URL("${props['url']}/services/Dashboard"))

dashboard.getMyProjects(props['user'], props['password']).each { proj->
   println "name: ${proj.name}, id: ${proj.name}"
   println "workflows:"
   dashboard.getProjectWorkflows(props['user'], props['password'], proj.id).each { wf->
       println "\t${wf.name}"
   }
}
```

# Create Your Own

Web services are great because the standard interface allows any team to create its own application using the technologies of their choice. As an example, here is a small Ruby program that looks up the **`getRecentWorkflowActivity` operation**:

```
require 'soap/wsdlDriver'
include SOAP

wsdl_file = 'http://localhost:8080/services/Dashboard?wsdl'
username = ARGV[0]
```

```
password = ARGV[1]

# Load WSDL and create driver
factory = SOAP::WSDLDriverFactory.new(wsdl_file)
ws = factory.create_rpc_driver
# ws.wiredump_dev = STDOUT

activities = ws.getRecentWorkflowActivity(username, password, 10)

activities.each { |activity|
  printf("Retrieved activity: %s %s %s %s %s\n",
         activity.buildLifeId,
         activity.projectName,
         activity.workflowName,
         activity.latestStamp,
         activity.latestStatus)
}
```

**And building a project is just as easy:**

```
require 'soap/wsdlDriver'
include SOAP

wsdl_file = 'http://localhost:8080/services/Dashboard?wsdl'
username = ARGV[0]
password = ARGV[1]
projectId = ARGV[2]
workflowId = ARGV[3]
force = ARGV[4]

# Load WSDL and create driver
factory = SOAP::WSDLDriverFactory.new(wsdl_file)
ws = factory.create_rpc_driver
# ws.wiredump_dev = STDOUT

ws.buildProject(username, password, projectId, workflowId, force)
```

# Operations Supported by the Interface

Most of the operations supported by this interface are used to lookup recent build-life activity. The exceptions are the operations that run originating and non-originating workflows. The operations supported are detailed below.

## getProjectRequest

AnthillPro responds to this request by looking up the appropriate project and returning a ProjectFacade for it. This is used to lookup the name for projects.

- **Parameters:**
  - Username
  - Password
  - Project Id

- **Response (getProjectResponse):**
  - ProjectFacade

# getWorkflowServerGroupsRequest

Requests a list of ServerGroups that this user may run a given workflow on.

- **Parameters:**
  - Username
  - Password
  - Workflow Id
- **Response (getWorkflowServerGroupsResponse):**
  - An array of ServerGroupFacades (ArrayOfServerGroupFacade)

# getRecentWorkflowActivityRequest

Requests a listing of the most recent workflows that have been executed.

- **Parameters:**
  - Username
  - Password
  - Count. An integer detailing the number of workflow activity items requested.
- **Response (getRecentWorkflowActivityResponse):**
  - An array of WorkflowActivityFacades (ArrayOfWorkflowActivityFacade)

# getRecentActivityForProjectRequest

Requests a listing of the most recent workflows that have been executed for a given project.

- **Parameters:**
  - Username
  - Password
  - Count. An integer detailing the number of workflow activity items requested.

- **Response (getRecentActivityForProjectResponse):**

- An array of WorkflowActivityFacades (ArrayOfWorkflowActivityFacade)

# getRecentActivityForWorkflowRequest

Requests a listing of the most recent workflows that have been executed for a given workflow.

- **Parameters:**
  - Username
  - Password
  - Workflow Id
  - Count. An integer detailing the number of workflow activity items requested.
- **Response (getRecentActivityForWorkflowResponse):**
  - An array of WorkflowActivityFacades (ArrayOfWorkflowActivityFacade)

# getMyProjectsRequest

Requests a listing of projects a user has access to.

- **Parameters:**
  - Username
  - Password

- **Response(getMyProjectsResponse):**
  - An array of ProjectFacades (ArrayOfProjectFacade)

# runWorkflowRequest

Triggers the specified non-originating workflow to run. To use this, you must identify the build life the workflow should be run against as well as the server group to run the workflow on.

- **Parameters:**
  - Username
  - Password
  - Build Life Id
  - Workflow Id

- Environment Id

- **Response(runWorkflowResponse):**
  - This response is just an acknowledgement that the request has been received.

## buildProjectRequest

Triggers the specified originating workflow to run for the specified project. The force flag instructs it to continue running even if there are no changes detected in source control.

- **Parameters:**
  - Username
  - Password
  - Project Id
  - Workflow Id
  - Force Build (boolean)

- **Response (buildProjectResponse):**
  - This response is just an acknowledgement that the request has been received.

# Available Objects

All data retrieved through the SOAP interface will be exposed as one of the following objects. There are a set of provided Java classes that model these items. Of course, they can be modeled using the tools of your choice.

## WorkflowFacade. A named workflow for a project.

| Field Name | Type | Description |
|---|---|---|
| id | String | The internal id for the workflow. |
| name | String | The name of the workflow. |
| originating | Boolean | True if the workflow is originating. |
| projectId | String | The internal id of the project. |

## WorkflowActivityFacade. The record of an executed workflow.

| Field Name | Type | Description |
|---|---|---|
| buildLifeId | String | The internal id for the build life. |
| caseId | String | The internal id of the workflow case. |
| date | String | The date the workflow was run. |
| latestStamp | String | The most recent stamp applied to this build life. |
| latestStatus | String | The most recent status attained by this build life. |
| projectId | String | The internal id of the workflow executed. |
| ProjectName | String | The name of the project the workflow was executed on. |
| workflowId | String | The internal id of the workflow executed. |
| workflowId | String | The human readable name of the workflow executed. |

# ServerGroupFacade. The name and id of a server group.

| Field Name | Type | Description |
|---|---|---|
| id | String | The internal id for a server group. |
| name | String | The name of a server group. |

# ProjectFacade. The name and id of a project.

| Field Name | Type | Description |
|---|---|---|
| id | String | The internal id for a project. |
| name | String | The name of a project. |

## Note

The WSDL exposes each of these object types collected in the form of an array as well.