

The modules are used to provide various information to the user. All enabled modules are displayed in the top menu below the *Search* input. The current version of ADEI by default implements:

- *Wiki?* module provides user-defined information about the system and preview charts
- *Graph* module is a main ADEI component providing data plots and easy navigation capabilities
- *Control* module provides access to the control information

It is possible to register new modules which will provide additional setup-specific information. First, the name of the module should be defined. Then, it is necessary to create a module initialization file (*module\_name.php*) in ADEI *modules/* subdirectory. This file should include 3 mandatory methods (the *module* prefix must be replaced with the module name. For example, for alarms module instead of *\$module\_title*, the *\$alarms\_title* should be used):

- The variable *module\_title* contains the module name (as it should be shown to the users in menu):

```
$module_title = _("Title of the Module")
```

- The function *moduleJS* contains JavaScript? initialization code
- The function *modulePage* contains HTML code of the page

Since, the modules are normally dynamically updated during the ADEI execution. The real HTML context should not be set by *modulesPage* functions, but provided using services. The *moduleJS* function should initialize JavaScript? object which will take care of context updating and return the name of this object.

When the module is ready it should be enabled in the configuration file *config.php* (or in the appropriate *setup*).

The ADEI includes several JavaScript? classes providing standard use-cases. They are described below.

## XML Module

This module uses an *ADEI service* generating XML content and XSLT stylesheet to update information on the page. The *modulePage* function should just define a single *div* element:

```
<div id="module_div" class="xml_module">Loading...</div>
```

The *moduleJS* function initializes new XMLMODULE object supplying the id of div created in *modulePage* function:

```
module_object = new XMLMODULE("module_div");
```

and returns the name of created object:

```
return "module_object";
```

Complete example of a module (alarms module):

```
$alarms_title = _("Alarms");  
function alarmsJS() {  
    echo 'alarms = new XMLMODULE("alarms_div");';  
    return "alarms";  
}
```

```

}
function alarmsPage() {
    echo '<div id="alarms_div" class="xml_module">Loading...</div>';
}

```

By default, the service *module\_name.php* is used to obtain XML document and stylesheet *module\_name.xslt* is used to convert XML into the HTML. The service should reside in the *services/* subdirectory (read about creating services [here](#)). The XSLT stylesheet should be placed in the *xslt* folder. However, it is possible to have more elaborated service configuration. For that it is necessary to define special *update* service. This service should be created in *services/update* directory and be named *module\_name.php*. It should contain a single function, called *ADEIServiceGetUpdateInfo*, which accept a *REQUEST* object as a parameter and return associative array with two elements:

- *xml* - specifying the full URL of the ADEI service along with all properties (the current options could be obtained and modified using *GetQueryString?* method of *REQUEST* object, see example bellow)
- *xslt* - specifying the name of XSLT stylesheet (without extension)

The function is executed on each update and, therefore, can dynamically select the appropriate service depending on some of request parameters.

Example, the *alarms* update service (it uses *control.php* service to get XML data and adds two additional parameters to the *REQUEST* properties):

```

function ADEIServiceGetUpdateInfo(REQUEST $req) {
    if ($req->CheckData()) {
        $query = $req->GetQueryString($extra = array(
            "target" => "alarms_summary",
            "time_format" => "text"
        ));

        return array(
            "xml" => "services/control.php?$query",
            "xslt" => "alarms"
        );
    }
    return false;
}

```