



## Customizing iTop 0.9

*WORD OF CAUTION : version 0.9 was the last significant “beta” step before the “production” 1.0 release. Considering that starting with 1.0 the backward compatibility will be a must, the transition from 0.9 to 1.0 was the last chance to cleanup a certain number of things in the internals of iTop. Therefore the migration from 0.9 to 1.0 will NOT be transparent, especially for customized versions. If you consider doing significant customizations to iTop, we strongly encourage you to first migrate to iTop 1.0 beta before doing your customizations.*

*Also the customization (and the maintenance of your customized version) will be easier in version 1.0, because the structure of the application is clearer. (menus templates have been moved out of the database, the basic capabilities of the iTop “generic” user interface have been enhanced to integrate most of the features that were performed by the templates in 0.9, customizations can be stored as separate modules, etc...).*

### **1 What can be customized in iTop**

The iTop application was designed to be fully customizable. The data model of the application is fully configurable. The user interface is "generic" and adapts automatically to the underlying data model. There are two kind of things you can customize in iTop : display templates or the data model. Customize display templates if you want to adjust the appearance of objects (add images, add an IFRAME to integrate with an external application...)

Adapt the data model if you want to add attributes to the existing objects or create your own objects from scratch.

### **2 Customizing display templates**

iTop uses some special templates for two types of display:

- menus items
- display of any object's details

Both types of templates are based on HTML syntax with 4 iTop specific tags : 'itopblock', 'itoptabs', 'itoptab', 'itoptoggle'.

ITOPBLOCK

Syntax:

`<itopblock type="format">OQL Query</itopblock>`

This tag is used to display the result of a query in different formats

"list": suitable for a query returning (possibly) multiple results. Will be displayed as a list.

"bare\_details": suitable for a query returning 1 object. Displays the attributes (properties) of the object as a plain table.

"search": displays a search form with the same default values as the query

"links": used to display objects "linked" to another object (generally the current object, in a "details" page).

There are 2 useful placeholders, that can use to build the template of an object:

\$pkey\$: the ID of the current object

\$class\$: the class of the current object

Look at the examples in the directory business/templates

## ITOPTAB AND ITOPTABS

Syntax:

```
<itoptabs>
  <itoptab name="Tab_Display_Name">Whatever content for the tab</itoptab>
</itoptabs>
```

These tags are used to display a set of tabs. Of course you can include `<itopblock>` tags inside an `<itoptab>` tag.

## ITOPTOGGLE

Syntax:

```
<itoptoggle name="name of toggle" open="true|false">Content to be toggled</itoptoggle>
```

This tag is used to create a « toggle » to expand/collapse some part of the web page, like for the display of the « last modification » of an object.

See the templates in business/templates for examples of usage.

## MENU TEMPLATES

Menu templates use basically the same syntax: a mix of HTML and itoptags; the only differences are:

- There is no current object for the menus, so \$pkey\$ and \$class\$ cannot be used.
- The templates are stored directly in the database, in the priv\_menuNode table (in the column "template").

## 3 Data Model

The iTop Data Model is based on PHP classes. Each class of object manipulated by iTop corresponds to a valid PHP class.

In order to be stored/maintained in the MySQL database by the iTop "ORM" (Object Relational Mapper) layer, the definition of such classes uses a specific API, provided by the static "MetaModel" class.

In version 0.9 most classes are defined in the files located in the "business" directory. For example most of the Configuration Items are defined in the file "itop.business.class.inc.php" in this directory.

To adapt the data model to your needs you can either modify these definitions or create your own classes from scratch.

In both cases follow the process below:

1. Modify the PHP classes definitions
2. Point your browser to the "toolkit.php" page, at the root of the iTop directory.
3. Click on the link "Check model, Create DB, Update DB (new class, new attribute)".
4. Click on the link "Biz model consistency".
5. Adjust your data model to fix all the errors reported. Run this link again to double-check.
6. When there is no more inconsistency in your model, click on the link "Concordance between Biz model and DB format" and follow the instructions to synchronize your database with the data model.

When you're done you can use the iTop application with your new/modified data model, create new instances of your objects, edit them, etc...

A class definition occurs, for each class, inside the method Init. This definition defines 5 types of information:

1. Basic information about the class: name, label, table to use in the database, template for the display...
2. The list of Attributes (i.e. members of the class)
3. The list of Search Criteria for the class. (i.e. which attribute will be search-able - note that in iTop 1.0 this list is obsolete, all attributes are search-able by default).
4. The state machine for this class, if the class is state based (for example for Workflow Tickets).
5. Lists (called Zlists) defining the appearance of the object inside list of results or details...

Check the numerous examples in the "business" directory (especially `itop.business.class.inc.php` and `incidentMgmt.business.php`) for examples of what can be done with an iTop object.