ERP5 performance testing

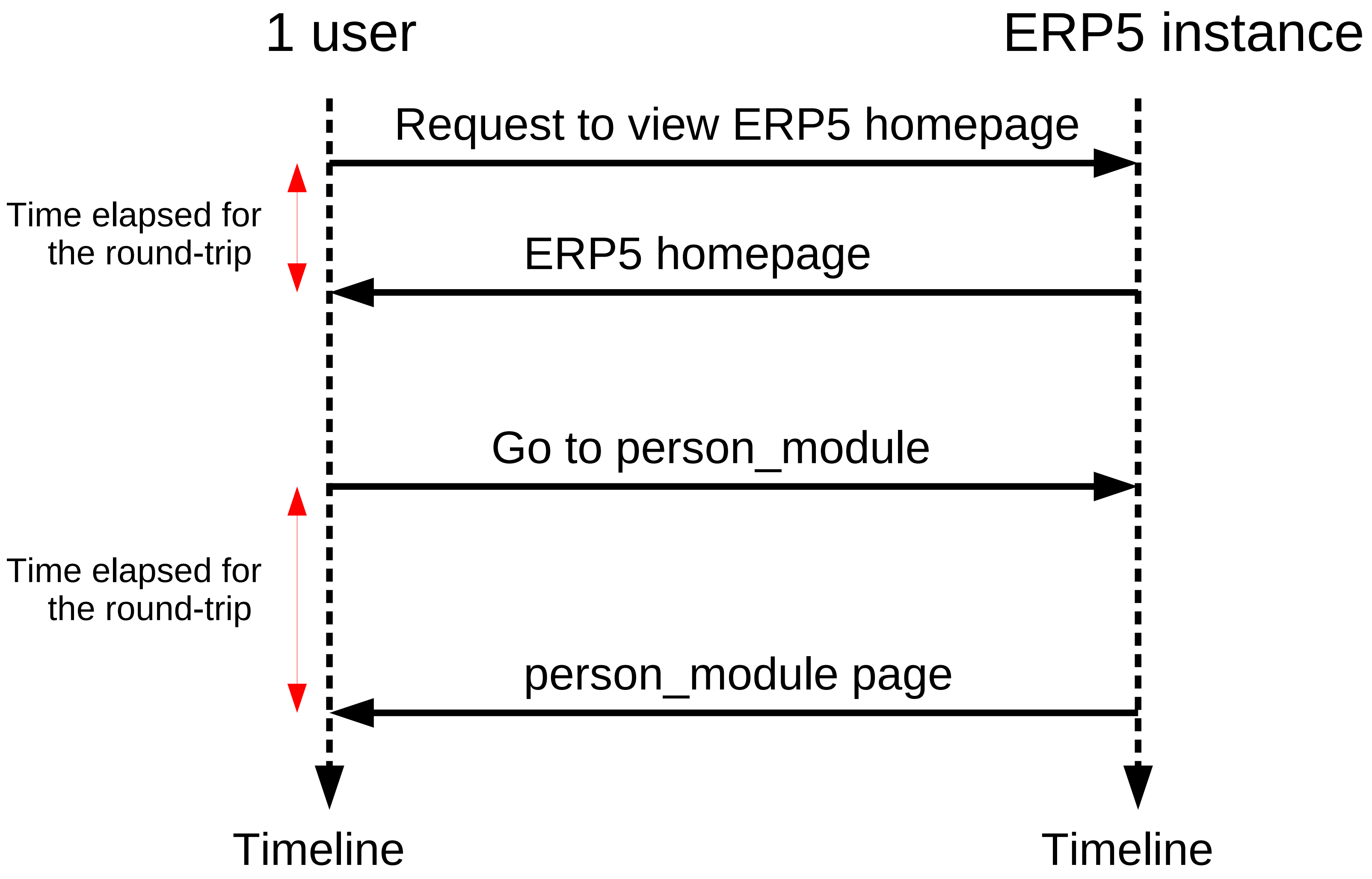
This guide will teach you:

- How to perform performance testing
- How to write scripts suitable for performance testing for ERP5

Support ERP5 !
www.erp5.org/howtohelp

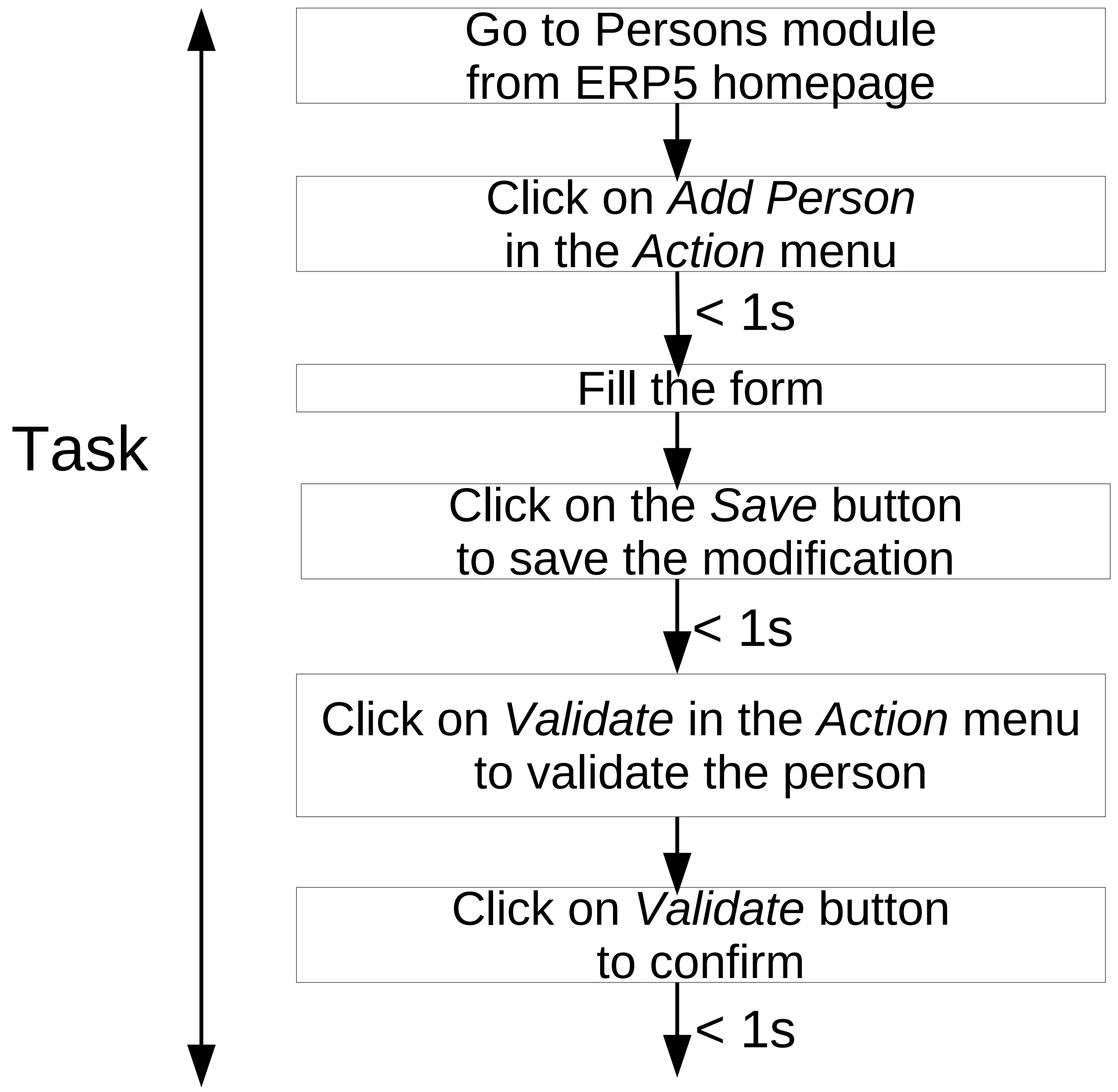
© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Performance testing


1 user
ERP5 instance
Request to view ERP5 homepage
ERP5 homepage
Go to person_module
person_module page
Time elapsed for the round-trip
Time elapsed for the round-trip
Timeline
Timeline

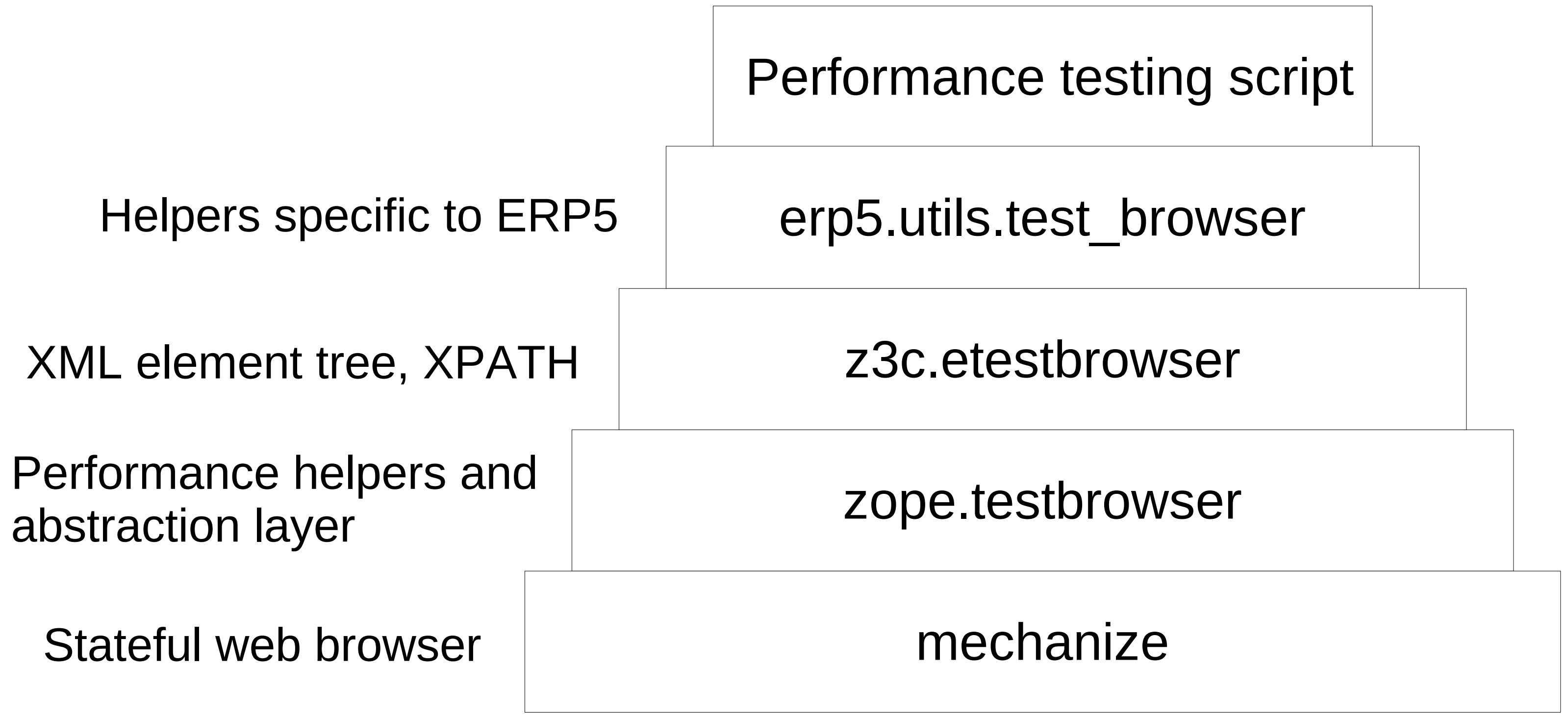
© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Detailed performance test plan


Task
Go to Persons module from ERP5 homepage
Click on *Add Person* in the *Action* menu
< 1s
Fill the form
Click on the *Save* button to save the modification
< 1s
Click on *Validate* in the *Action* menu to validate the person
Click on *Validate* button to confirm
< 1s

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

ERP5 performance testing framework



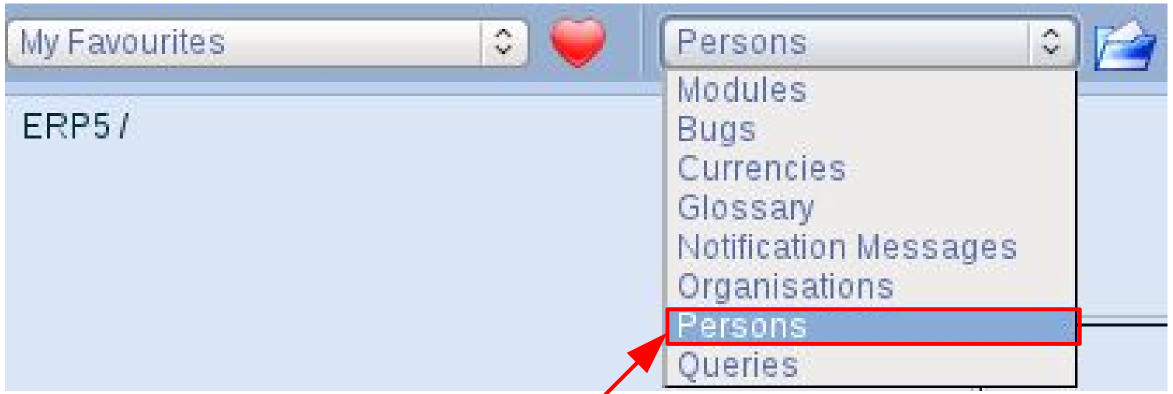
© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Basic script

```
from erp5.utils.test_browser.browser import Browser

def benchmarkAddPerson():
  """Benchmark adding a person"""
  # Create a browser instance
  browser = Browser('http://foo/', 'erp5', username='zope', password='zope')

  # Open ERP5 homepage
  browser.open()

if __name__ == '__main__':
  benchmarkAddPerson()
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Go to *Persons* module from ERP5 homepage



```
<option value="/erp5/person_module">Persons</option>
```

value

label

By value

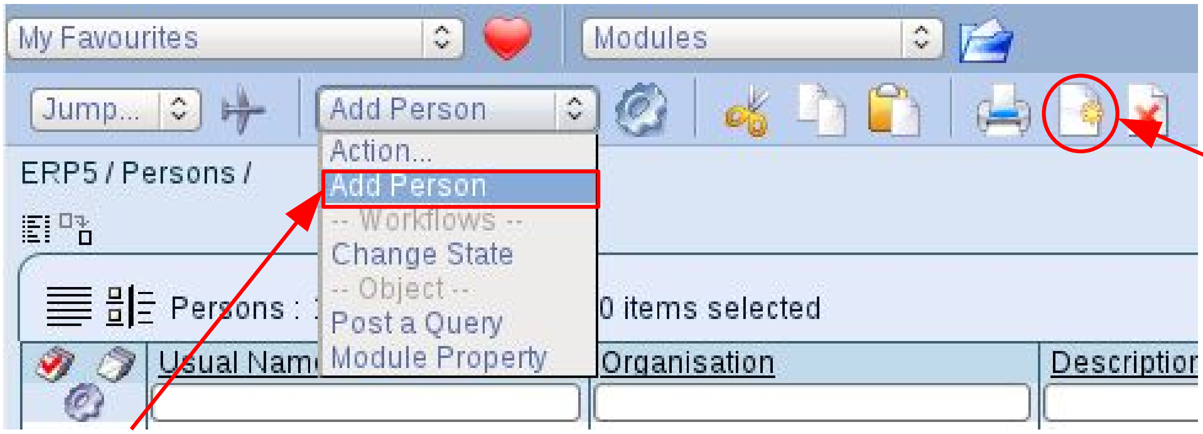
```
browser.mainForm.submitSelectModule(value='/person_module')
```

By label

```
browser.mainForm.submitSelectModule(label='Persons')
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Click on *Add Person* in the *Action* menu



```
<option value="add Person">Add Person</option>
```

By value
```
browser.mainForm.submitSelectAction(value='add')
```

By label
```
browser.mainForm.submitSelectAction(label='Add Person')
```

New button
```
browser.mainForm.submitNew()
```

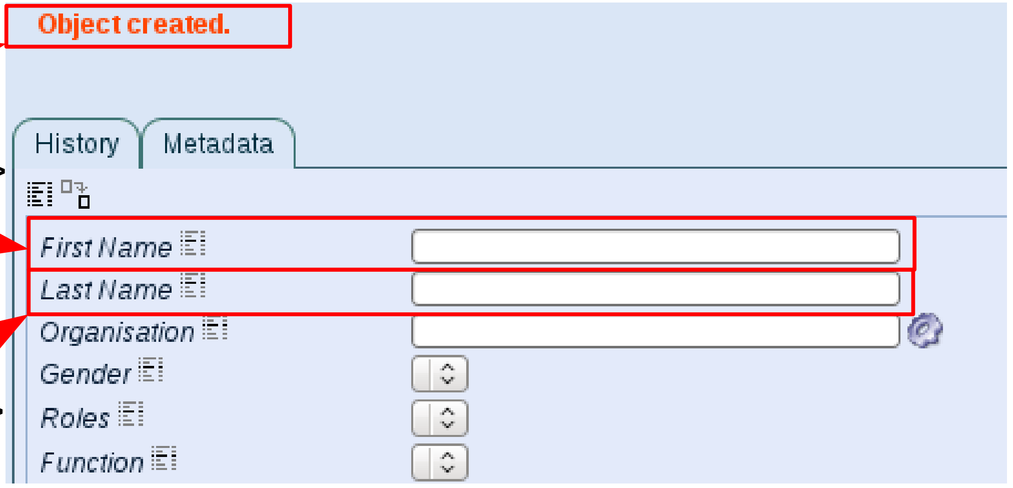
© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Fill the form

Transition message

<input name="field_my_first_name" value="" type="text" />

<input name="field_my_last_name" value="" type="text" />


Object created.

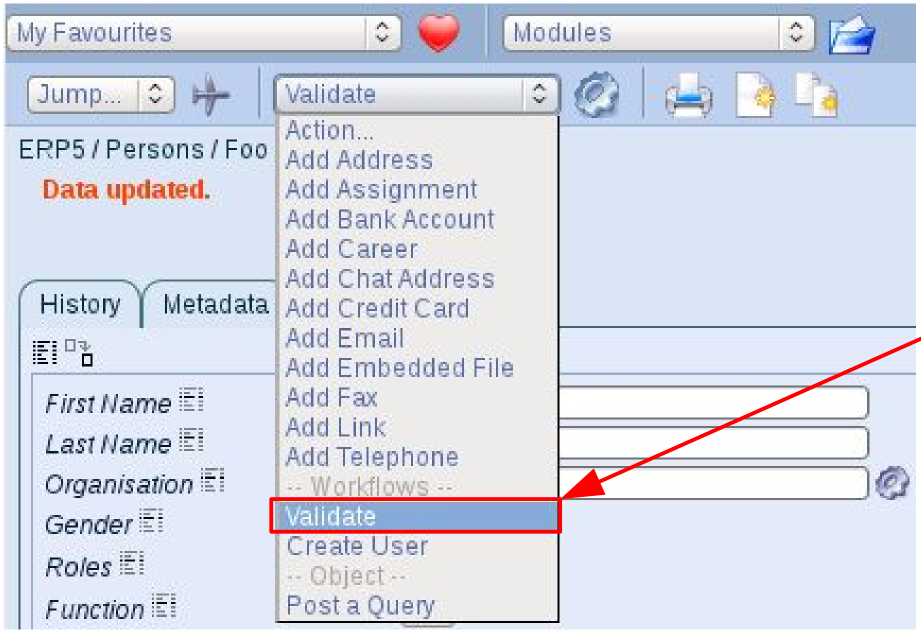
```
# Assert that the object has actually been created
assert browser.getTransitionMessage() == 'Object created.'

# Fill the First Name and Last Name fields
browser.mainForm.getControl(name='field_my_first_name').value = 'Foo'
browser.mainForm.getControl(name='field_my_last_name').value = 'Bar'
```

Click on the *Save* button to save the modification

```
# Save the modification
browser.mainForm.submitSave()

# Assert that the object has actually been saved
assert browser.getTransitionMessage() == 'Data updated.'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Click on *Validate* in the *Action* menu to validate the person



```
<option value="...Base_viewWorkflowActionDialog
?workflow_action=validate_action">Validate
</option>
```

value

label

By value

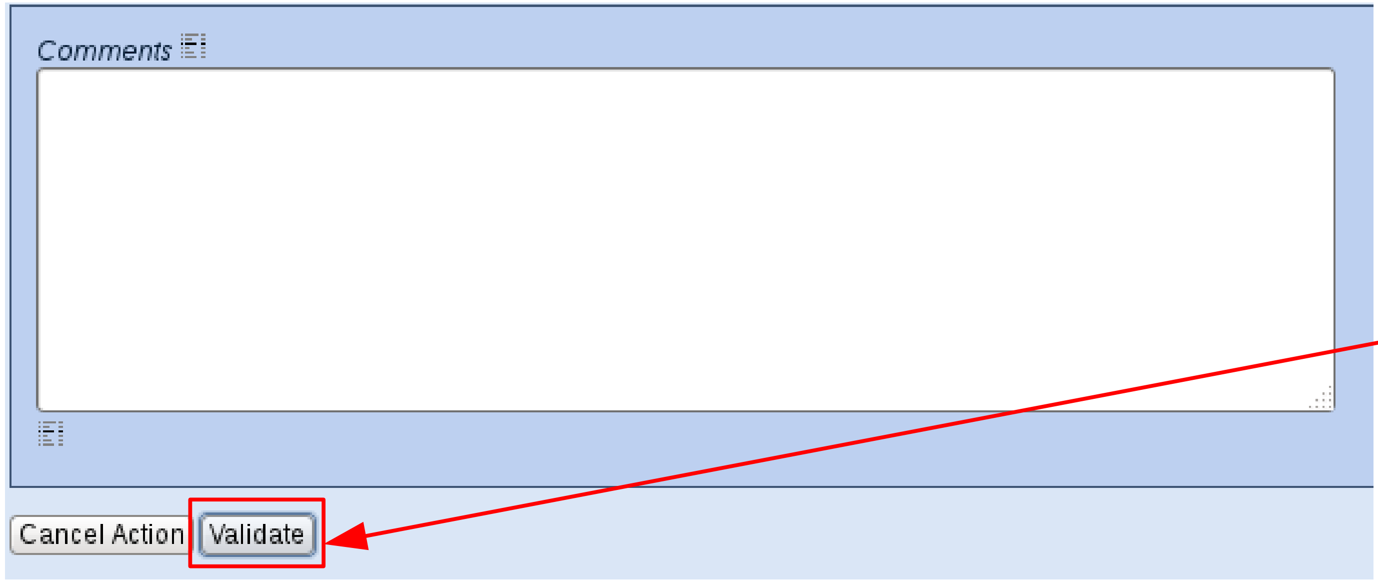
```
browser.mainForm.submitSelectWorkflow(value='validate_action')
```

By label

```
browser.mainForm.submitSelectWorkflow(label='Validate')
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Click on *Validate* button to confirm



Confirm dialog button

```
# Validate the person
browser.mainForm.submitDialogConfirm()

# Assert that the object has actually been validated
assert browser.getTransitionMessage() == 'Status changed.'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Measuring performances (1)

```
from erp5.utils.test_browser.browser import Browser

ITERATION = 20

def benchmarkAddPerson(result_dict):
  browser = Browser('http://foo/', 'erp5', username='zope', password='zope')
  browser.open()

  browser.mainForm.submitSelectModule(value='/person_module')

  result_dict.setdefault('Create', []).append(browser.mainForm.timeSubmitNewInSecond())
  assert browser.getTransitionMessage() == 'Object created.'

  browser.mainForm.getControl(name='field_my_first_name').value = 'Foo'
  browser.mainForm.getControl(name='field_my_last_name').value = 'Bar'
  result_dict.setdefault('Save',
                         []).append(browser.mainForm.timeSubmitSaveInSecond())
  assert browser.getTransitionMessage() == 'Data updated.'

  browser.mainForm.submitSelectWorkflow(value='validate_action')
  result_dict.setdefault('Validate',
                         []).append(browser.mainForm.timeSubmitDialogConfirmInSecond())
  assert browser.getTransitionMessage() == 'Status changed.'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Measuring performances (2)

```
if __name__ == '__main__':
  result_dict = {}
  counter = 0
  while counter != ITERATION:
    benchmarkAddPerson(result_dict)
    counter += 1

  for title, time_list in result_dict.iteritems():
    print "%s: %.4fs" % (title, float(sum(time_list)) / ITERATION)
```

Results :

```
Create: 0.4259s
Save: 0.5297s
Validate: 0.4146s
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Further readings

- erp5.utils.test_browser API documentation

  See `README.txt` in the source code
- z3c.etestbrowser documentation

  http://pypi.python.org/pypi/z3c.etestbrowser
- zope.testbrowser documentation

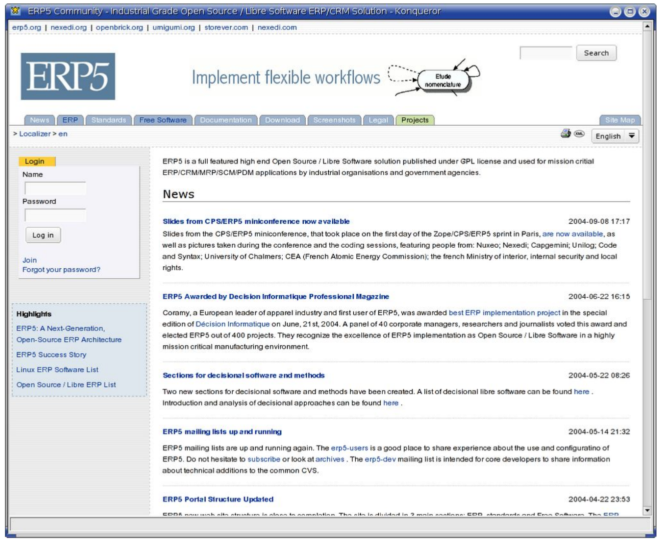
  http://pypi.python.org/pypi/zope.testbrowser

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Commercial Support

Enterprise Services

- Consulting
- Professional Training
- Custom Development


ERP5 Community - Industrial Grade Open Source / Libre Software ERP/CRM Solution - Konqueror
erp5.org | nexedi.org | openbrick.org | umigumi.org | storever.com | nexedi.com
Search
Implement flexible workflows
Etude nomenclature
News
ERP
Standards
Free Software
Documentation
Download
Screenshots
Legal
Projects
Site Map
> Localizer > en
English
Login
Name
Password
Log in
Join
Forgot your password?

ERP5 is a full featured high end Open Source / Libre Software solution published under GPL license and used for mission critial ERP/CRM/MRP/SCM/PDM applications by industrial organisations and government agencies.

News

Slides from CPS/ERP5 miniconference now available 2004-09-08 17:17

Slides from the CPS/ERP5 miniconference, that took place on the first day of the Zope/CPS/ERP5 sprint in Paris, are now available, as well as pictures taken during the conference and the coding sessions, featuring people from: Nuxeo; Nexedi; Capgemini; Unilog; Code and Syntax; University of Chalmers; CEA (French Atomic Energy Commission); the french Ministry of interior, internal security and local rights.

ERP5 Awarded by Decision Informatique Professional Magazine 2004-06-22 16:15

Coramy, a European leader of apparel industry and first user of ERP5, was awarded best ERP implementation project in the special edition of Décision Informatique on June, 21st, 2004. A panel of 40 corporate managers, researchers and journalists voted this award and elected ERP5 out of 400 projects. They recognize the excellence of ERP5 implementation as Open Source / Libre Software in a highly mission critical manufacturing environment.

Sections for decisional software and methods 2004-05-22 08:26

Two new sections for decisional software and methods have been created. A list of decisional libre software can be found here . Introduction and analysis of decisional approaches can be found here .

ERP5 mailing lists up and running 2004-05-14 21:32

ERP5 mailing lists are up and running again. The erp5-users is a good place to share experience about the use and configuratino of ERP5. Do not hesitate to subscribe or look at archives . The erp5-dev mailing list is intended for core developers to share information about technical additions to the common CVS.

ERP5 Portal Structure Updated 2004-04-22 23:53

ERP5 new web site structure is close to completion. The site is divided in 3 main sections: ERP, standards and Free Software. The ERP

Highlights

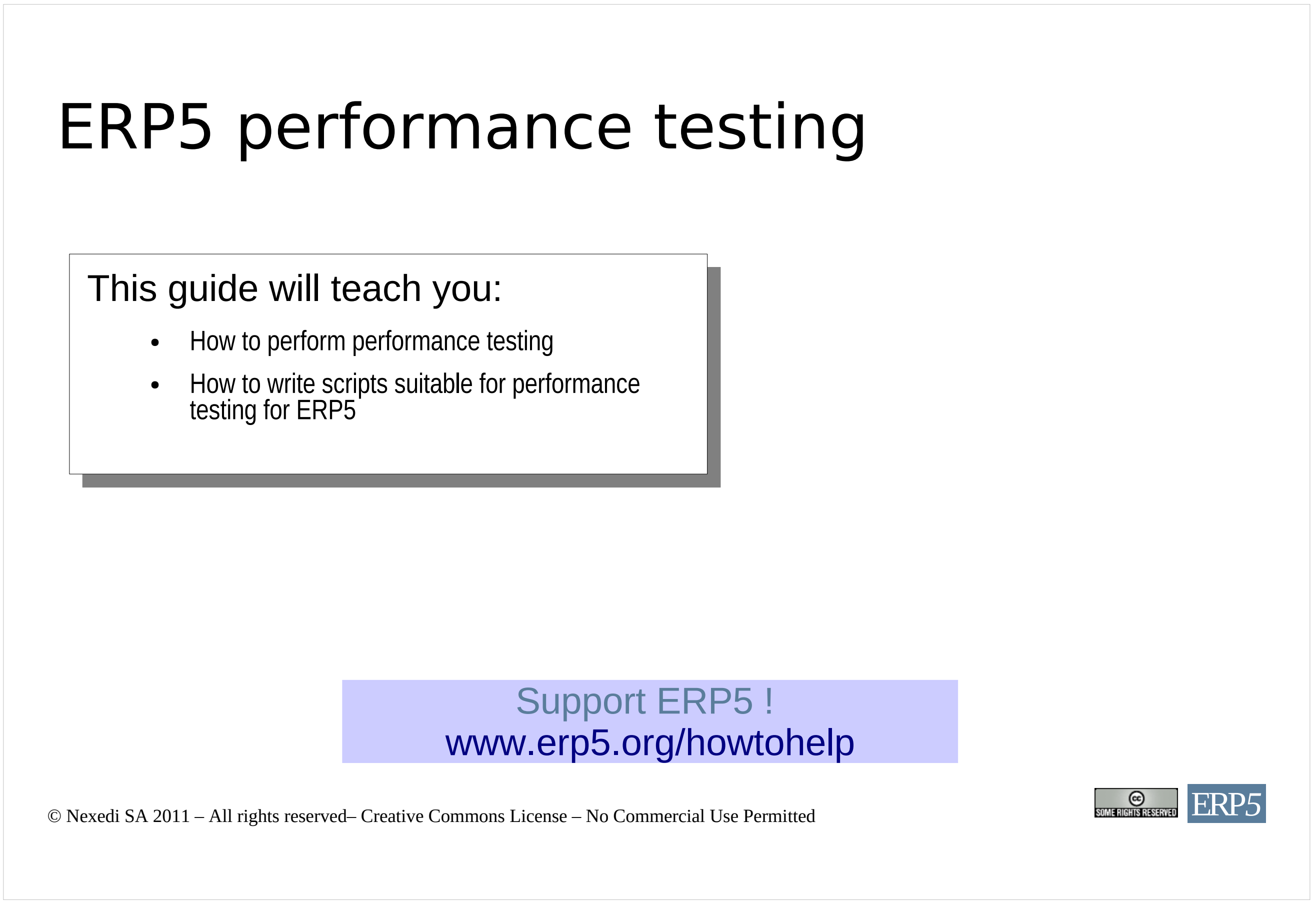
- ERP5: A Next-Generation, Open-Source ERP Architecture
- ERP5 Success Story
- Linux ERP Software List
- Open Source / Libre ERP List

www.erp5.org
Open Portal

Join ERP5 Service Network !
www.erp5.org/support

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

SOME RIGHTS RESERVED



ERP5 performance testing

This guide will teach you:

- How to perform performance testing
- How to write scripts suitable for performance testing for ERP5

Support ERP5 !
www.erp5.org/howtohelp

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

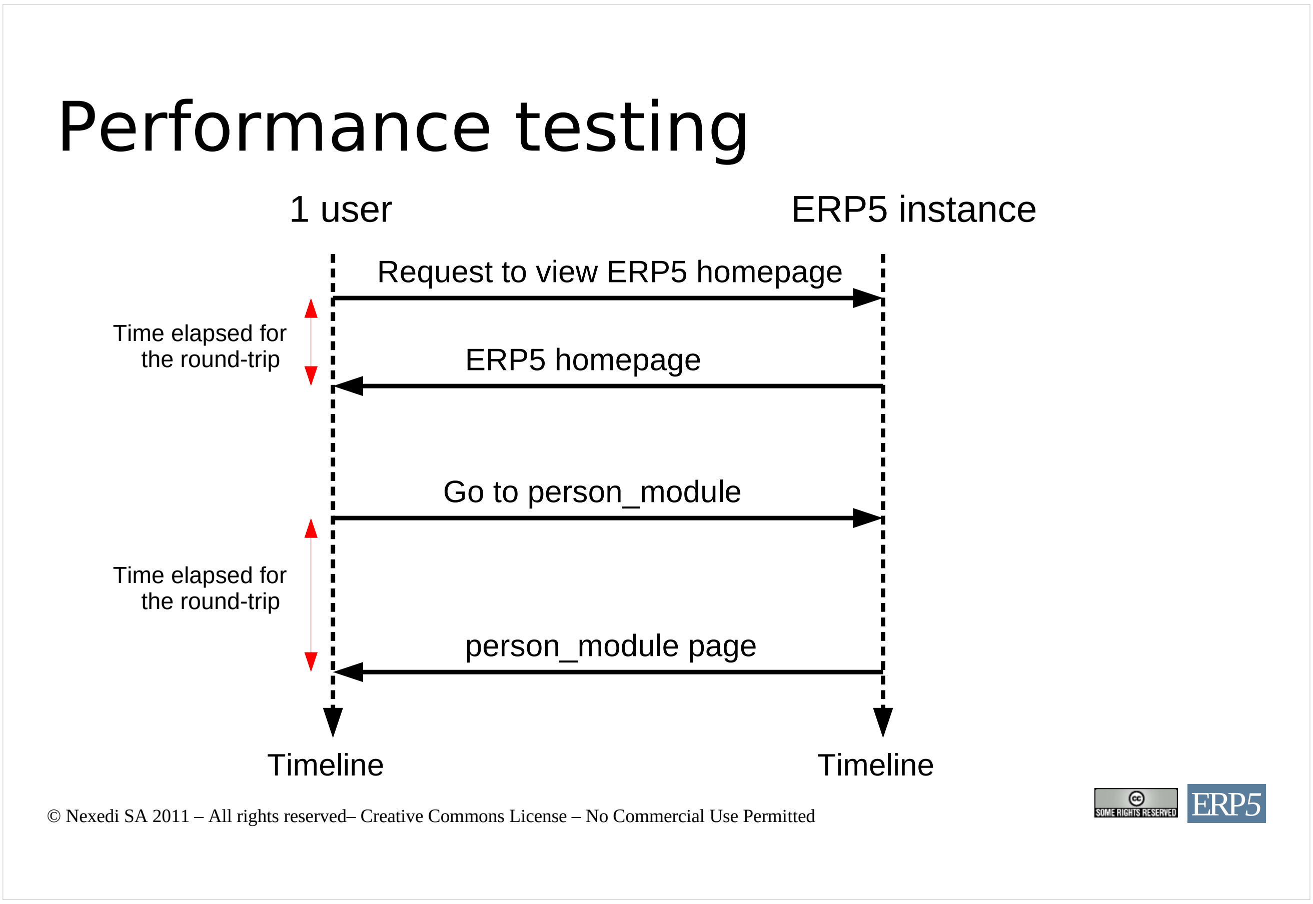
This visual guide has been created for learning and for teaching performance testing for ERP5. This visual guide is mostly useful to ERP5 developers who need to understand and performed performance testing, to users who are willing to understand ERP5 and to marketing people who need to explain performance testing for ERP5.

Readers should first have a quick look at the illustration on the upper part of each page then read the short text bellow the illustration carefully and associate each word written in bold to the corresponding item(s) in the illustration. For example, the term **Creative Commons License** is written in bold because it defines the license of the above illustration.

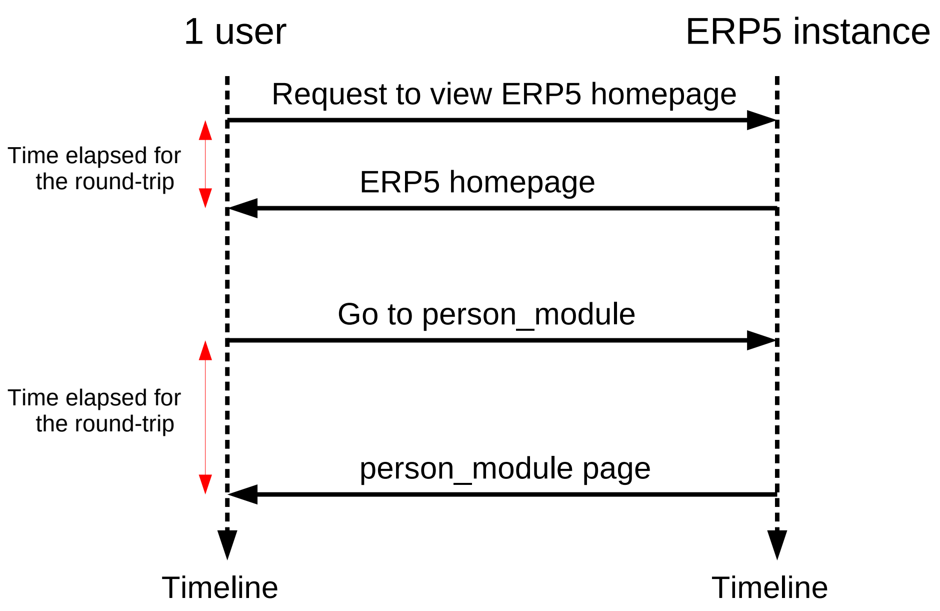
Copyright

You are free to copy, distribute, display, and perform the work under the following conditions: you must attribute the work in the manner specified by the author or licensor; you may not use this work for any commercial purposes including training, consulting, advertising, self-advertising, publishing, etc.; you may not alter, transform, or build upon this work.

For any reuse or distribution, you must make clear to others the license terms of this work. Any of these conditions can be waived if you get permission from the copyright holder through a commercial license or an educational license. For more information, contact info@nexedi.com



Performance testing

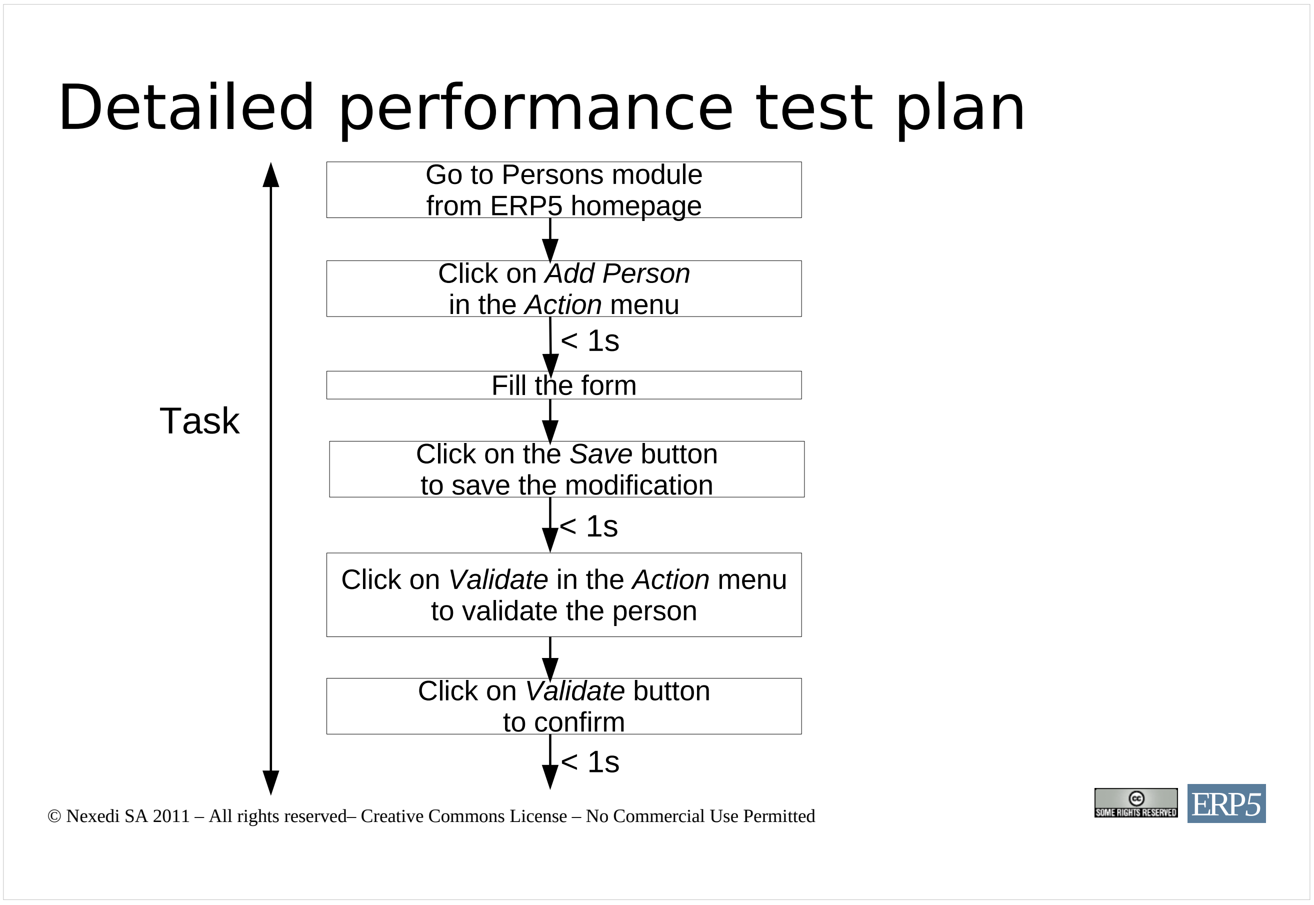

1 user
ERP5 instance
Request to view ERP5 homepage
Time elapsed for the round-trip
ERP5 homepage
Go to person_module
Time elapsed for the round-trip
person_module page
Timeline
Timeline

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Performance testing is about ensuring that the user gets the reply to his request in a timely fashion without considering concurrent accesses, and addressing bottlenecks. Once performance testing has been done, scalability testing is then performed to address potential bottlenecks arising with concurrent accesses. Scalability testing will be explained in a separate presentation.

When performing performance testing, first of all, a performance test plan has to be defined according to the most common tasks performed by users. Secondly, tasks must be group together by relevancy, where each operation of a group defines precisely realistic and acceptable times elapsed for the round-trip from an user point of view. Thirdly, a script must be written for each group gathering reproducible results. These script usually rely on an existing tool or framework.

This presentation will explain how to perform these three steps by giving a simple example and will rely on erp5.utils.test_browser Python package as the performance testing framework.



Detailed performance test plan

Task
Go to Persons module from ERP5 homepage
Click on *Add Person* in the *Action* menu
< 1s
Fill the form
Click on the *Save* button to save the modification
< 1s
Click on *Validate* in the *Action* menu to validate the person
Click on *Validate* button to confirm
< 1s
ERP5

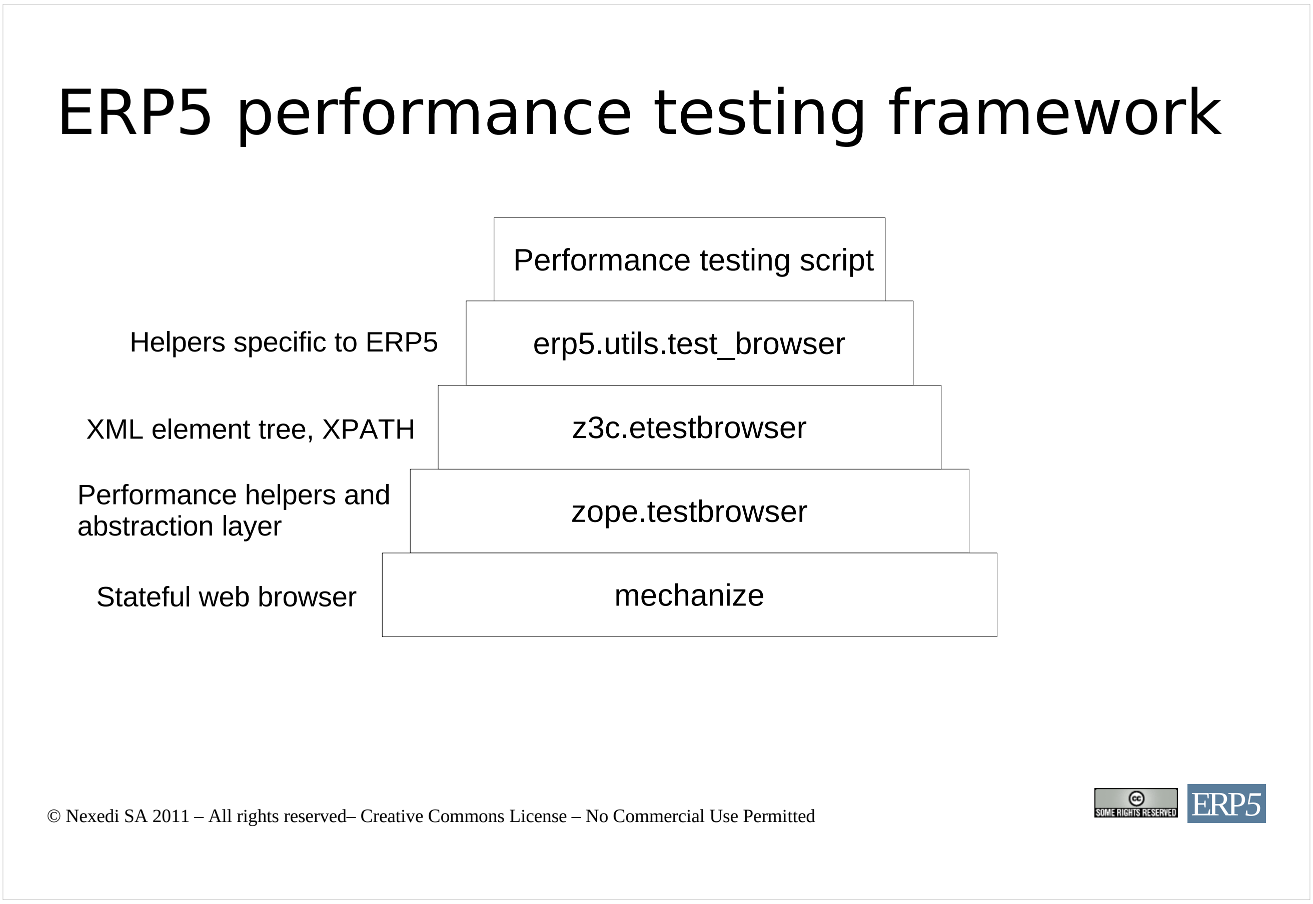
© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

In the following slides of this presentation, a simple example will be used where the user only add users. This task is the most common tasks performed in this example, therefore this is where the performances matters the most. As it is only composed of basic operation, they all should not be perceptible by the user.

In this example, there is only one group, namely adding persons, but generally there will be much more groups. For example, if managing bugs is another common tasks performed by users, there could be another group about adding, modifying and deleting bugs.

Along with the common tasks being performed for this example, there are also the steps done by the user to achieve such tasks. At this point, there is everything required to write the script to test the performance of these tasks.

The following slides will show how to write a script for this example after introducing briefly the performance testing framework written specifically for ERP5.



ERP5 performance testing framework

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Before explaining how to write the scripts for the performance test plan described before, it is necessary to introduce the framework which is going to be used for performance testing.

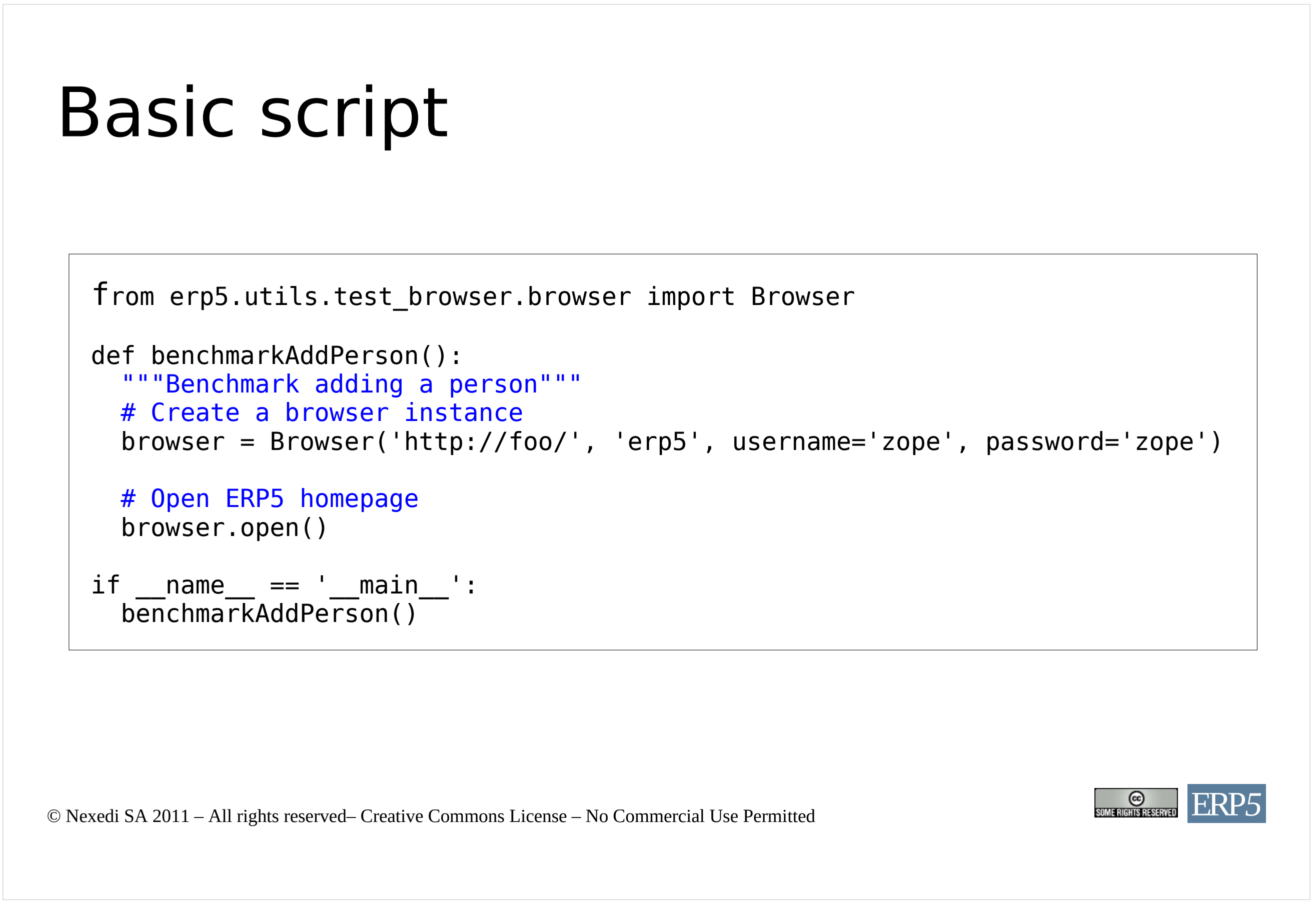
`mechanize` is a Python package providing a stateful programming web browser. Namely, it provides an API which allows to simulate what an user performs through a web browser by writing Python code. For example, it allows to open an URL, get the response, fill and submit forms, as well as supporting browser history and cookies.

The Python package `zope.testbrowser` is actually an abstraction of mechanize with a special focus on testing by measuring the elapsed time and pystones for each request. Besides of that, it also provides a nice and easy to use object model for forms and links.

The Python package `z3c.etestbrowser` extends `zope.testbrowser` by allowing to parse an HTML page as an elements tree and use XPATH for example to easily selects elements within an HTML or XML document. It also provides HTML and XML normalisation to display such contents in a readable way.

`erp5.utils.test_browser` only provides helpers specific to ERP5, such as selecting an ERP5 module, a favourite, copying and pasting previously selected objects, or getting the number of remaining activities.

However, note that Javascript is not supported (because it would require a Javascript engine), which could be circumvented by calling the target URL of Javascript code for example. Anyhow, Javascript testing is more related to functional testing and could be performed with Windmill or Selenium for example.



Basic script

```
from erp5.utils.test_browser.browser import Browser

def benchmarkAddPerson():
  """Benchmark adding a person"""
  # Create a browser instance
  browser = Browser('http://foo/', 'erp5', username='zope', password='zope')

  # Open ERP5 homepage
  browser.open()

if __name__ == '__main__':
  benchmarkAddPerson()
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

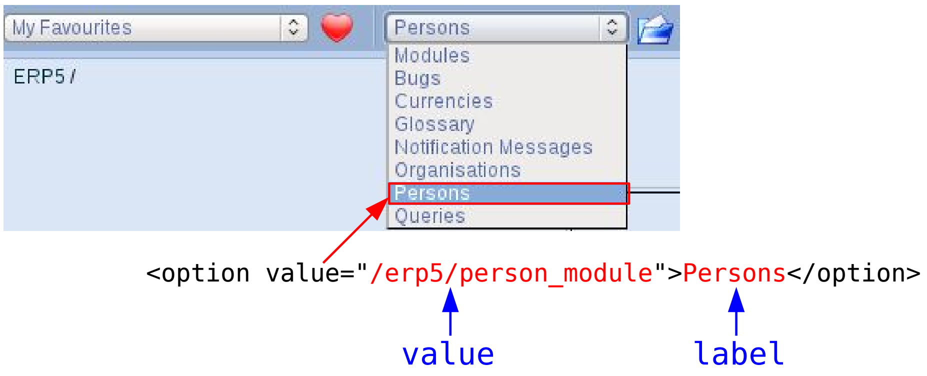
Before doing anything, you should have installed `erp5.utils.test_browser` package by running the following command line:

```
easy_install -f 'http://www.nexedi.org/static/packages/source/' erp5.utils.test_browser
```

This is just a basic script going to ERP5 homepage after instanciating `Browser` class from ERP5 benchmark module. The URL should always have a trailing slash. Note that you can give to `open()` method either a relative path to the ERP5 site (for instance, if you wish to open */erp5/person_module*, you just have to pass *person_module* to `open()`) or an absolute URL (the former is generally the most commonly used).

In the next slides, the function `benchmarkAddPerson()` will be completed following the steps given in the slide before.

Go to *Persons* module from ERP5 homepage



```
<option value="/erp5/person_module">Persons</option>
```

By value

```
browser.mainForm.submitSelectModule(value='/person_module')
```

By label

```
browser.mainForm.submitSelectModule(label='Persons')
```

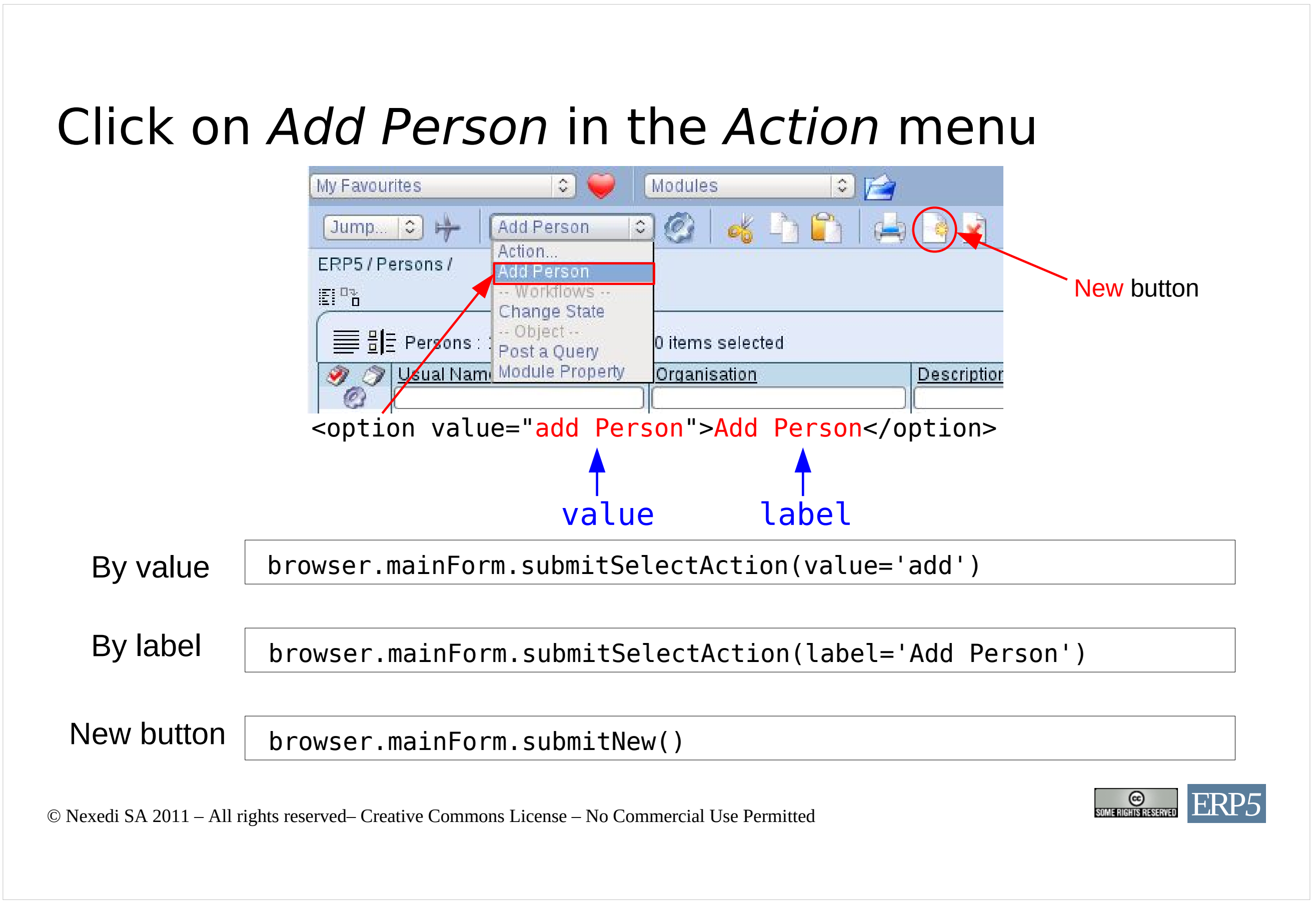
© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

In ERP5, there is only one form, whose *id* is `main_form`. Therefore, we first get the main `Form` object (as defined in `zope.testbrowser` package and extended by `erp5.utils.test_browser` package) and then select the module either by `value` or `label` (in terms of HTML, it means selecting the `<option>` by `value` or `label` within a `<select>` and can be therefore used the same way for any `<select>`).

`label` is of course easier to use as it does not required looking at the HTML source code, however, it is preferable to use the `value` as it will always remain the same whereas the `label` is more susceptible to change and is language-dependent.

Note that the given `label` value is searched as case-sensitive whole words within the option labels, and that the `value` is searched in the same way and also if it is found at the end of the string excluding the query string.

Internally, it will actually select the field and then submit it, therefore the URL before executing this code will be http://foo/erp5/ and http://foo/erp5/person_module/ after.



Click on *Add Person* in the *Action* menu

```
<option value="add Person">Add Person</option>
```

By value

```
browser.mainForm.submitSelectAction(value='add')
```

By label

```
browser.mainForm.submitSelectAction(label='Add Person')
```

New button

```
browser.mainForm.submitNew()
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

As explained in the previous slide, you can either select an `<option>` within a `<select>` by `value` or `label`. Furthermore, you could also use the `New` button.

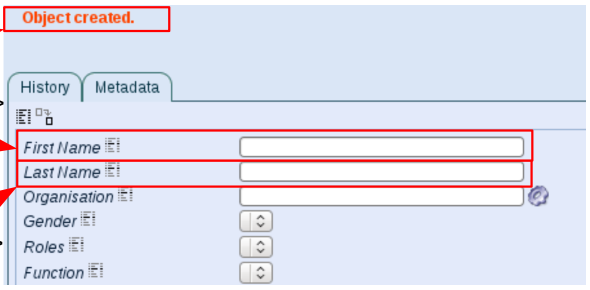
Similarly, most of the buttons and menus can be activated through methods defined in the main form instance (from `erp5.utils.test_browser` package), such as `Delete` and `Copy` button by calling respectively `submitDelete()` and `submitCopy()` or `Favourite` by calling `submitSelectFavourite()` (which takes the same parameters as other `submitSelect*()` methods).

Fill the form

Transition message

<input name="field_my_first_name" value="" type="text" />

<input name="field_my_last_name" value="" type="text" />

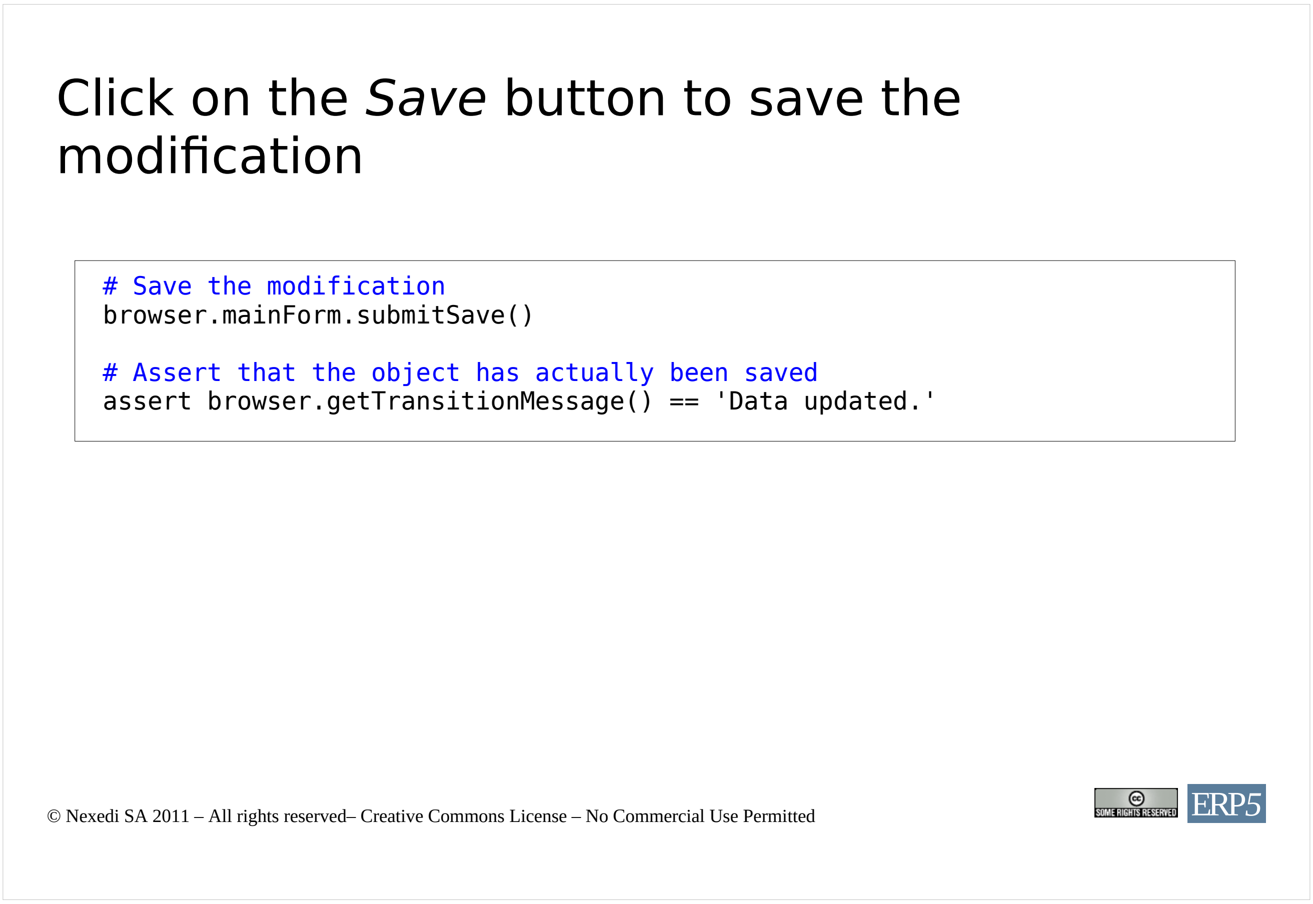


```
# Assert that the object has actually been created
assert browser.getTransitionMessage() == 'Object created.'

# Fill the First Name and Last Name fields
browser.mainForm.getControl(name='field_my_first_name').value = 'Foo'
browser.mainForm.getControl(name='field_my_last_name').value = 'Bar'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

First, we check whether the person has been created by looking at the transition message (also known as the Portal status message) by calling `getTransitionMessage()`.

Then we fill the fields for the first name and last name by getting the control by their name and setting the value which is going to be submitted. `getControl()` method could be used to get any field on the page either by name or label (the latter is usually easier to use, as shown before, but of course it does not really make sense to use it on a text `<input>` field).

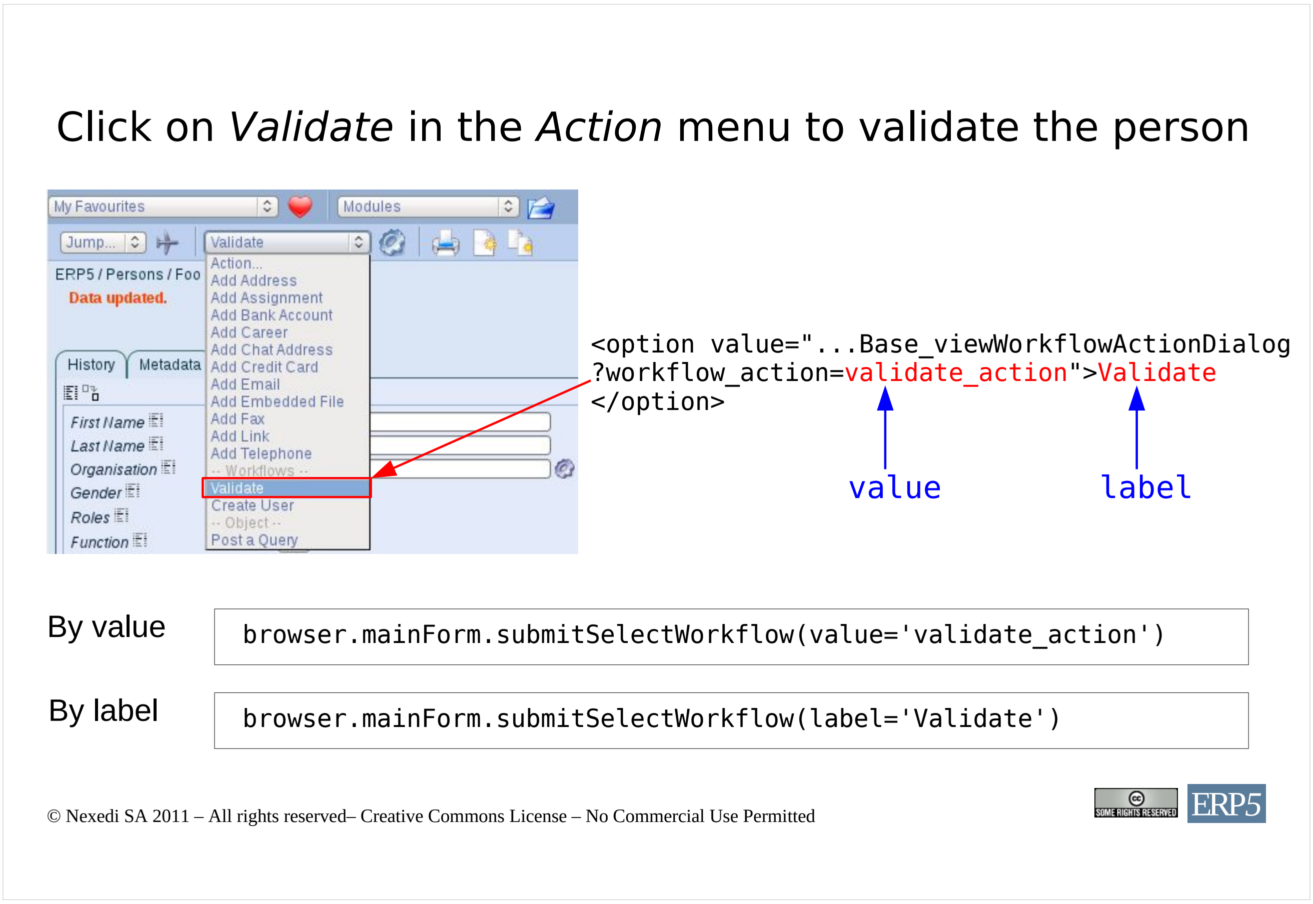


Click on the *Save* button to save the modification

```
# Save the modification
browser.mainForm.submitSave()

# Assert that the object has actually been saved
assert browser.getTransitionMessage() == 'Data updated.'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Similarly to the `New` button, we call `submitSave()` to save the modification made in the previous step and then check that the data has been actually saved properly.



Click on *Validate* in the *Action* menu to validate the person

```
<option value="...Base_viewWorkflowActionDialog
?workflow_action=validate_action">Validate
</option>
```

By value

```
browser.mainForm.submitSelectWorkflow(value='validate_action')
```

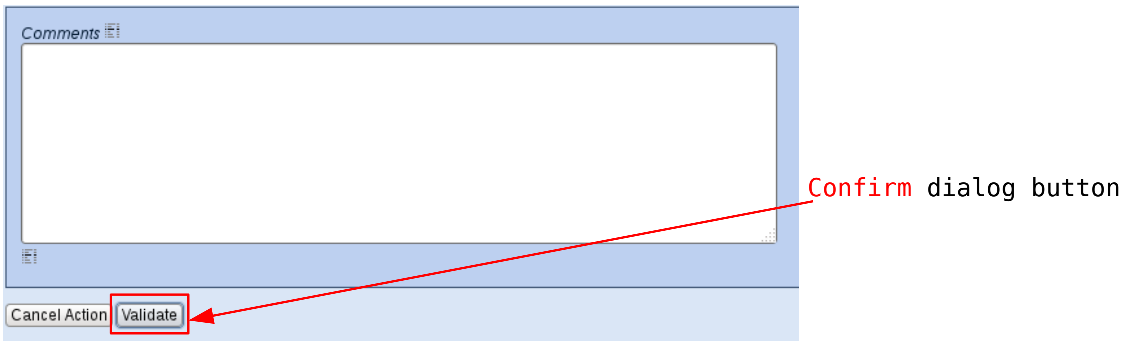
By label

```
browser.mainForm.submitSelectWorkflow(label='Validate')
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

`submitSelectWorkflow()` method is a bit different to other `submitSelect*()` methods for the `value` parameter because the workflow to be executed is given in the query string. The script identifier (`Base_viewWorkflowActionDialog` in this example) can also be given through the parameter `script_id` which defaults to `Base_viewWorkflowActionDialog`. However, the `label` parameter is the same as the other methods.

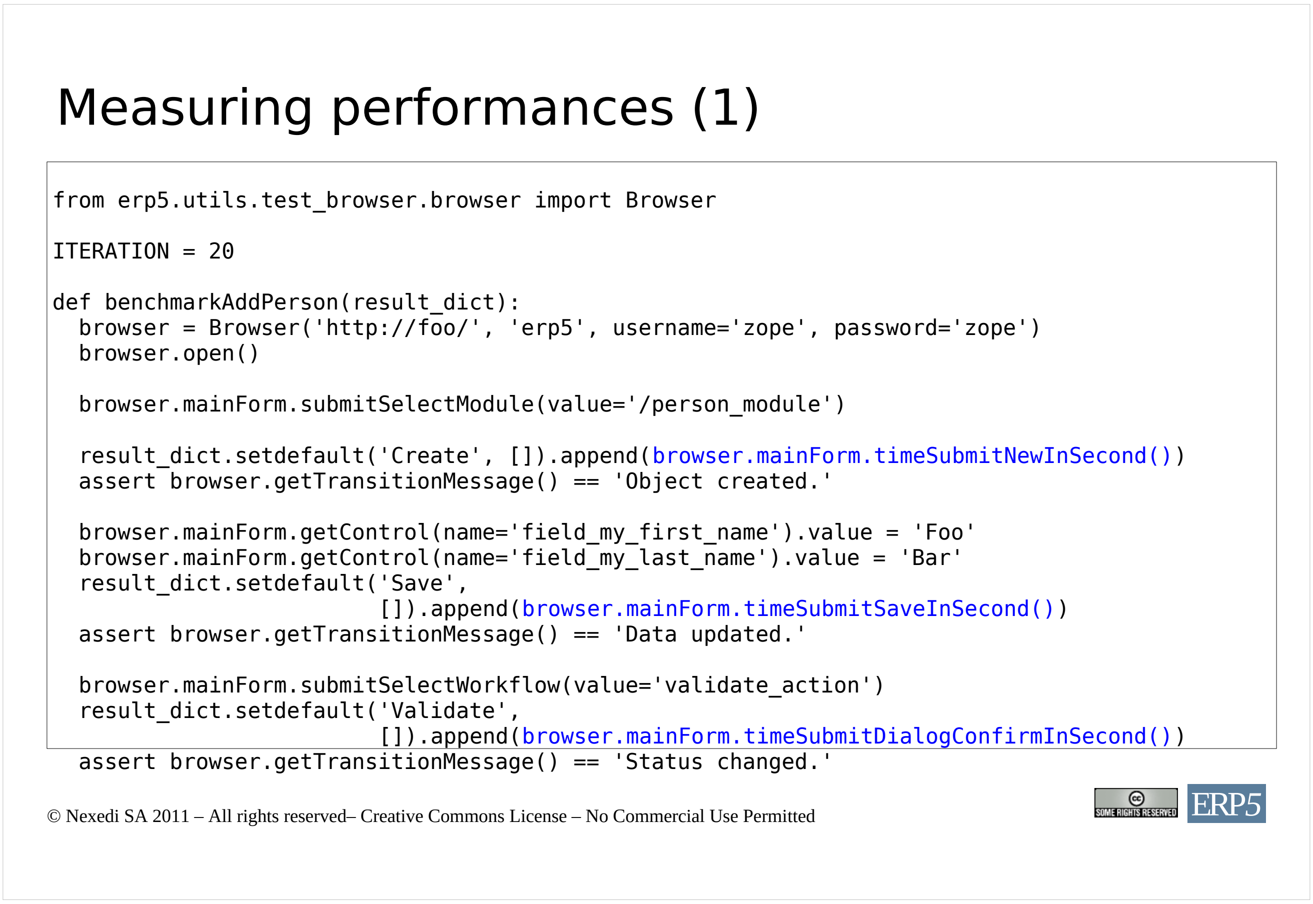
Click on *Validate* button to confirm



```
# Validate the person
browser.mainForm.submitDialogConfirm()

# Assert that the object has actually been validated
assert browser.getTransitionMessage() == 'Status changed.'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

Now confirm the dialog form by calling `submitDialogConfirm()`, there is also a `submitDialogCancel()`, and finally check that the validation is successful. After this point, it could also be relevant to check for the number of activities (by calling `getRemainingActivityCounter()` method) to ensure that it decreases after each call of the script main function.



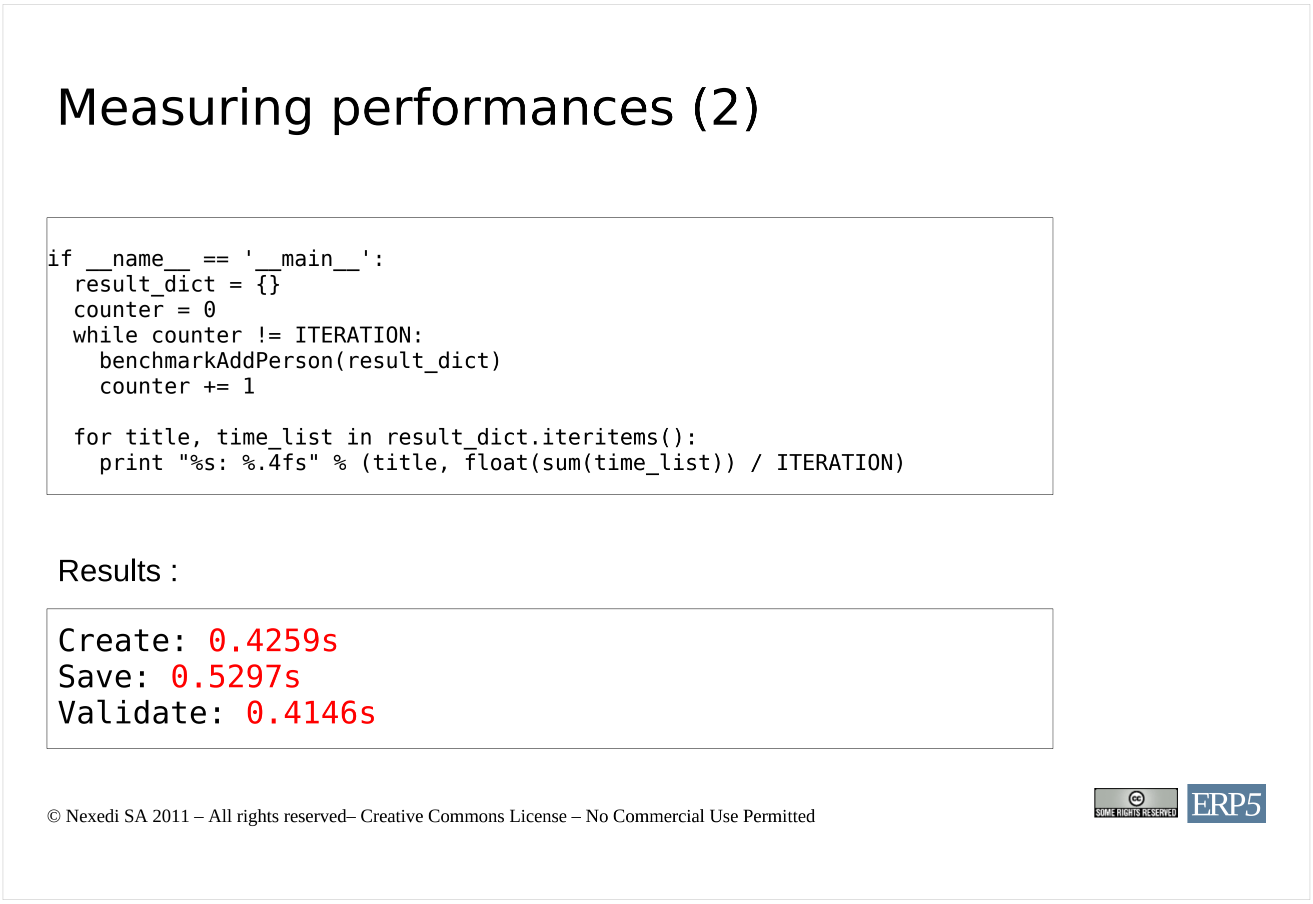
Measuring performances (1)

```
from erp5.utils.test_browser.browser import Browser

ITERATION = 20

def benchmarkAddPerson(result_dict):
  browser = Browser('http://foo/', 'erp5', username='zope', password='zope')
  browser.open()

  browser.mainForm.submitSelectModule(value='/person_module')

  result_dict.setdefault('Create', []).append(browser.mainForm.timeSubmitNewInSecond())
  assert browser.getTransitionMessage() == 'Object created.'

  browser.mainForm.getControl(name='field_my_first_name').value = 'Foo'
  browser.mainForm.getControl(name='field_my_last_name').value = 'Bar'
  result_dict.setdefault('Save',
                         []).append(browser.mainForm.timeSubmitSaveInSecond())
  assert browser.getTransitionMessage() == 'Data updated.'

  browser.mainForm.submitSelectWorkflow(value='validate_action')
  result_dict.setdefault('Validate',
                         []).append(browser.mainForm.timeSubmitDialogConfirmInSecond())
  assert browser.getTransitionMessage() == 'Status changed.'
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

This is the first part of the final script which shows how to measure the time elapsed upon creation, save and confirmation of the person. Basically, any method can be prefixed by `time` to get the number of seconds the request took.

As explained in a previous slide, it is essential to perform several iterations to get pertinent results. In this example, only 20 iterations will be done, but in real-world performance tests, there should be much more of course.



Measuring performances (2)

```
if __name__ == '__main__':
  result_dict = {}
  counter = 0
  while counter != ITERATION:
    benchmarkAddPerson(result_dict)
    counter += 1

  for title, time_list in result_dict.iteritems():
    print "%s: %.4fs" % (title, float(sum(time_list)) / ITERATION)
```

Results :

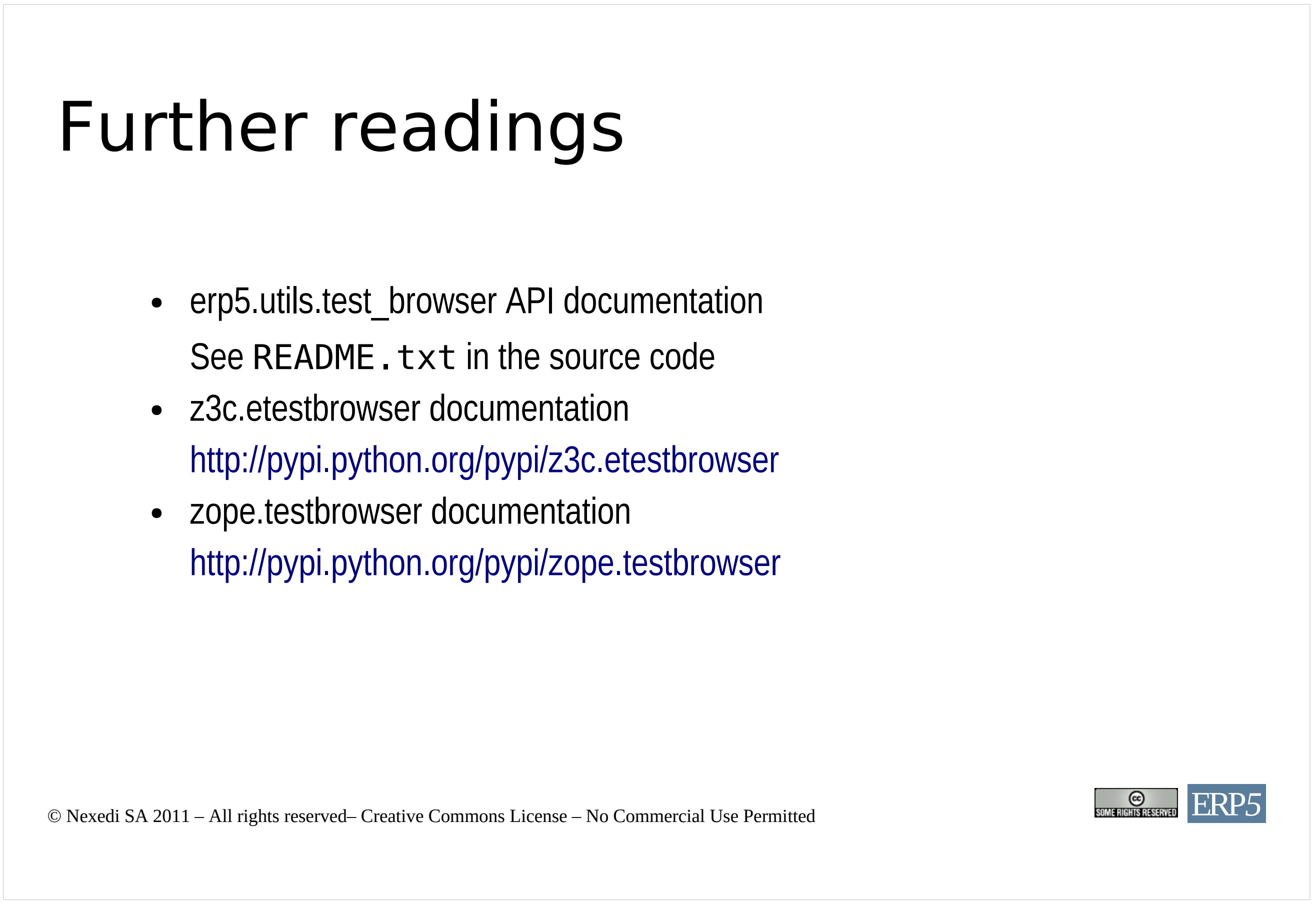
```
Create: 0.4259s
Save: 0.5297s
Validate: 0.4146s
```

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

The second part of the final script calls the main function 20 times and gathers the results. For each operation (create, save and validate), the average time is calculated and finally displayed. The final results shows that the average of each operation is approximatively half the acceptable timeframes defined at the beginning.

Shortcomings:

- If there are much more iterations, perhaps you should add a `sleep()` after each call as there might be many activities left which could slow down the next iteration.
- The average value may not be enough to measure the performances, therefore other statistic values could be calculated (such as standard deviation).
- Measuring the performance in seconds is fine when running the performance script on the same machine but for scalability testing, it will not be enough as it is dependant on the processor being run, so using pystones will be more appropriate.



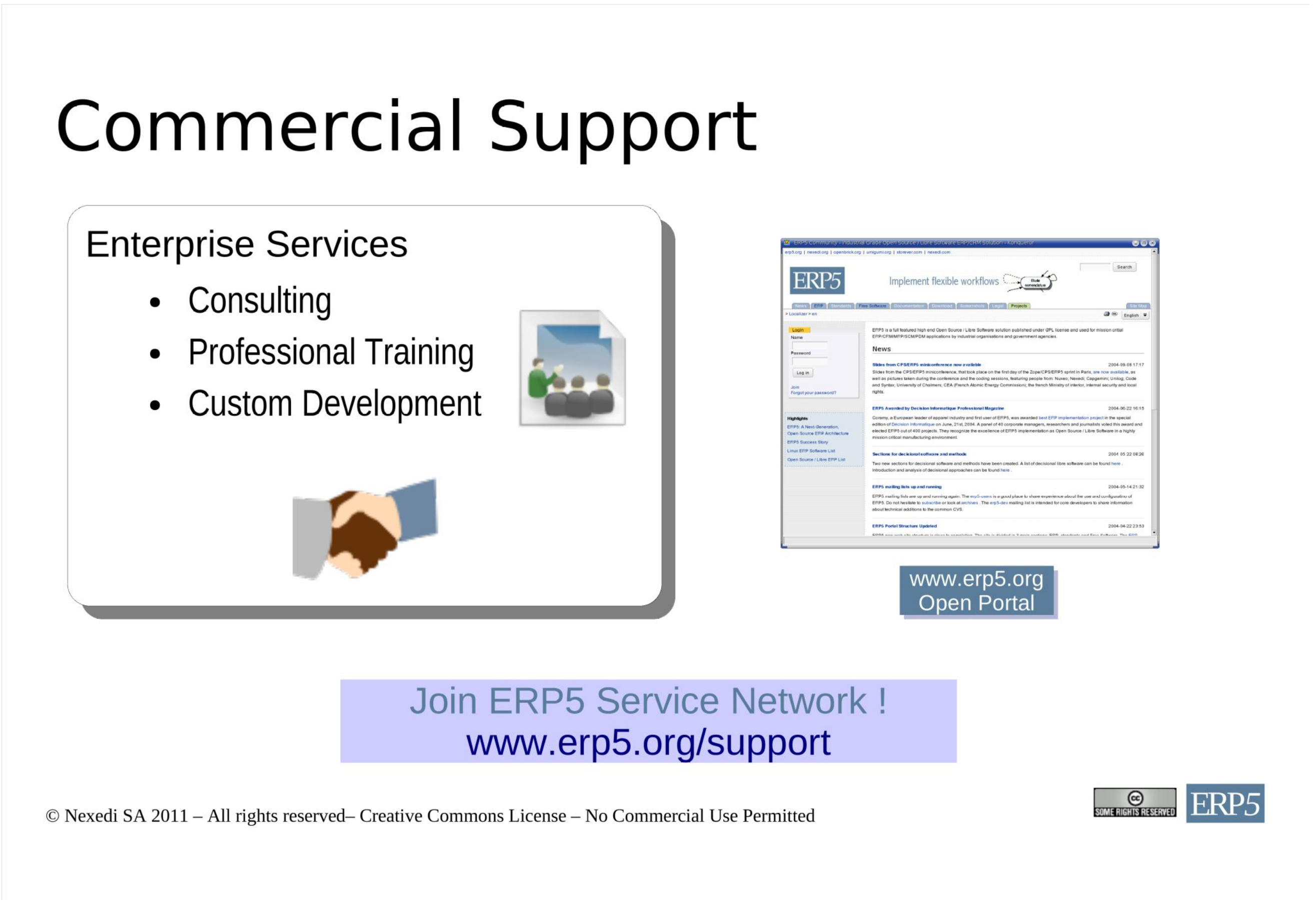
Further readings

- erp5.utils.test_browser API documentation
  See `README.txt` in the source code
- z3c.etestbrowser documentation
  http://pypi.python.org/pypi/z3c.etestbrowser
- zope.testbrowser documentation
  http://pypi.python.org/pypi/zope.testbrowser

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

You can download a tarball of `erp5.utils.test_browser` package on Nexedi website: http://www.nexedi.org/static/packages/source/. You can then generate the documentation as explained in `README.txt`. There is also an example in examples/ directory.

`z3c.testbrowser` and especially `zope.testbrowser` are extensively documented with examples on their respective project pages, and, as `erp5.utils.test_browser` follows exactly the same API, it is worth having a look.



Commercial Support

Enterprise Services
- Consulting
- Professional Training
- Custom Development

www.erp5.org
Open Portal

Join ERP5 Service Network !
www.erp5.org/support

© Nexedi SA 2011 – All rights reserved– Creative Commons License – No Commercial Use Permitted

The ERP5 project is coordinated by Nexedi. Certified Consulting and Development services are provided worldwide by a network of partner companies.

Just like any ERP, ERP5 is complex system which requires experience in order to provide complete satisfaction to clients. Configuring ERP5 requires much more effort and experience than simply installing ERP5 software components alone.

Companies interested in joining ERP5 professional services network are encouraged to contact Nexedi. Nexedi provides complete training and certification services. Nexedi can assist ERP5 service providers in providing optimal presales, implementation and aftersales service to enterprise clients. Nexedi is looking forward to assist consulting or software companies to build a sustainable, profitable and independent business.

For more information, please refer to www.erp5.org/support.