

# Selenium Design Patterns: Beyond the Page Object


Selenium Conference 2014  
Sept. 5, 2014

Derrick Kearney  
Purdue University  
telldsk@gmail.com

# Agenda

- WebForm Pattern
- ItemList Pattern
- IframeWrap Pattern

# Working with Web Forms



## Submit a Support Ticket

<b>Username:</b>	OPTIONAL	<b>Problem:</b>	REQUIRED
<input type="text"/>		<input type="text"/>	
<b>Name:</b>	REQUIRED		
<input type="text"/>			
<b>E-mail:</b>	REQUIRED		
<input type="text"/>			
<b>Please answer the question:</b>	REQUIRED	<b>Attach a screenshot:</b> OPTIONAL	
Sharks live in...		Browse... No file selected.	
<input type="text" value="- Select an answer -"/>		(.jpg, .jpeg, .jpe, .bmp, .tif, .tiff, .png, .gif, .pub, .xml, .zip, .mp4, .pdf, .doc, .txt)	
		<input type="button" value="Submit"/>	

# WebForm Pattern

Standardize the interface for filling out a web form.

## Usual Interface

```
po = TroubleReportForm()
```

```
po.set_name('testuser')  
po.set_email('tu@hubzero.org')  
po.set_problem('test problem')  
po.set_upload('myscreenshot.png')
```

```
po.submit.click()
```

## Proposed Interface

```
po = TroubleReportForm()
```

```
data = {  
    'name' : 'testuser',  
    'email' : 'tu@hubzero.org',  
    'problem' : 'test problem',  
    'upload' : 'myscreenshot.png',  
}  
po.populate_form(data)
```

```
po.submit_form()
```

# 1. Give page objects a 'value'

The screenshot shows a web form titled "Submit a Support Ticket" with a close button (X) in the top right corner. On the left, there are "Support Options" including "Knowledge Base", "Ask the Community", "Wish List", and "Support Tickets". The main form area contains the following fields:

- Username:** OPTIONAL, text input field.
- Name:** REQUIRED, text input field containing "testuser". This field is highlighted with a red dashed border.
- E-mail:** REQUIRED, text input field containing "tu@hubzero.org".
- Problem:** REQUIRED, text area containing "test problem".
- Attach a screenshot:** OPTIONAL, with a file input showing "myscreenshot.png" and a "Browse..." button. Supported formats: (.jpg, .jpeg, .jpe, .bmp, .tif, .tiff, .png, .gif).
- Please answer the question:** REQUIRED, text input field. The question is "Eye, ankle or arm: which is part of the head?".

A "Submit" button is located at the bottom right of the form.

```
...  
e = browser.find_element_by_id('trName')  
e.clear()  
e.send_keys("testuser")  
...
```

```
...  
name = Text('#trName')  
name.value = 'testuser'  
...
```

# Setting the value of a Checkbox

## Usual Interface

*# get the value*

```
e = browser.find_element_by_id('ckbox')  
e.is_selected()
```

*# set the value*

```
e = browser.find_element_by_id('ckbox')  
if e.is_selected() is not val:  
    e.click()
```

## Proposed Interface

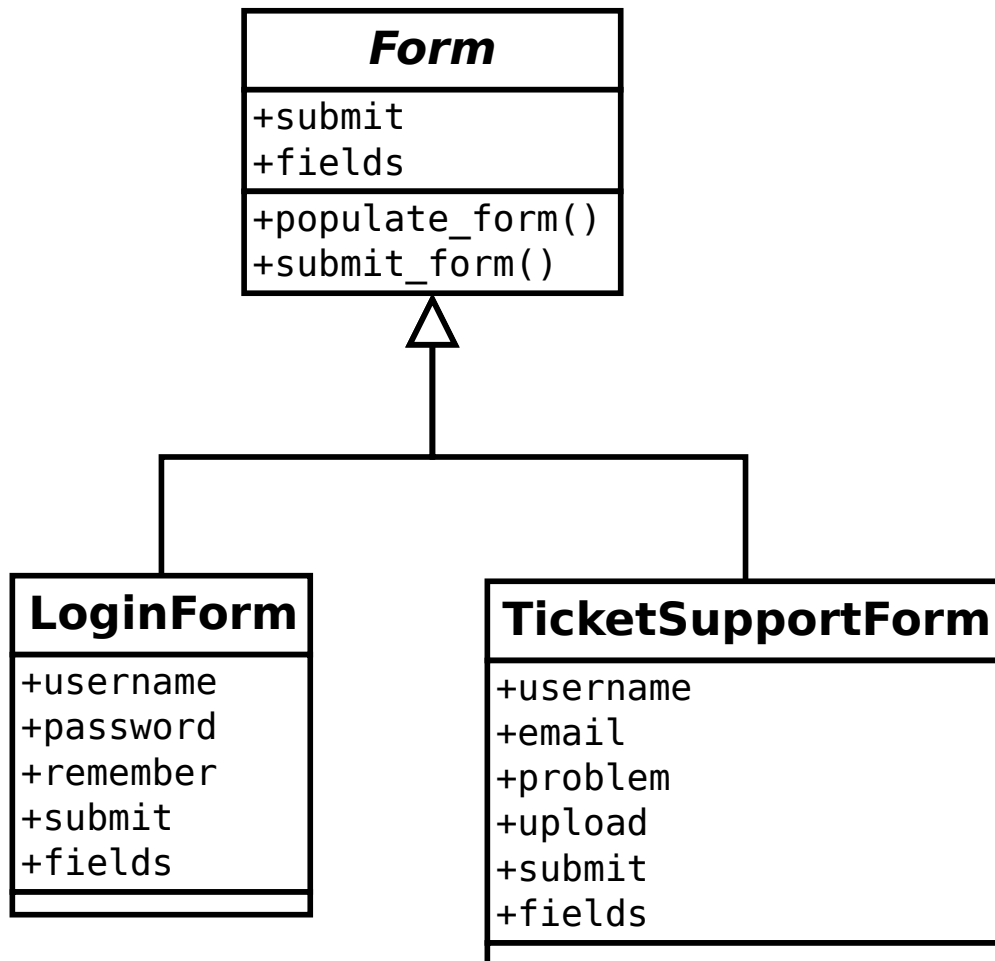
*# get the value*

```
e = Checkbox('#ckbox')  
x = e.value
```

*# set the value*

```
e = Checkbox('#ckbox')  
e.value = True
```

## 2. Use an abstract *Form* base class



## 2. Use an abstract *Form* base class

```
class LoginForm(object):
    def __init__(self):
        ...

    def set_username(self, username):
        ...

    def set_password(self, username):
        ...

    def login_as(self, username, passwd):
        ...
```

Log in with your Hub account.

Username:

[Forgot your Username?](#)

Password:

[Forgot your Password?](#)

☐ Remember Me

Login



## 2. Use an abstract *Form* base class

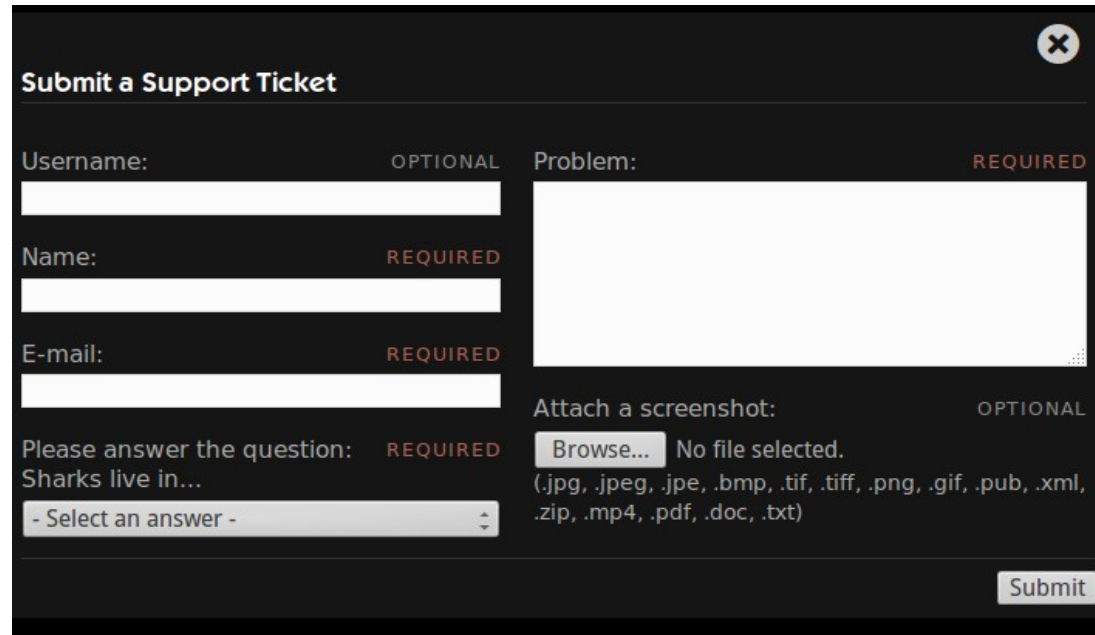
```
class TroubleReportForm(object):
    def __init__(self):
        ...

    def set_username(self, username):
        ...

    def set_name(self, name):
        ...

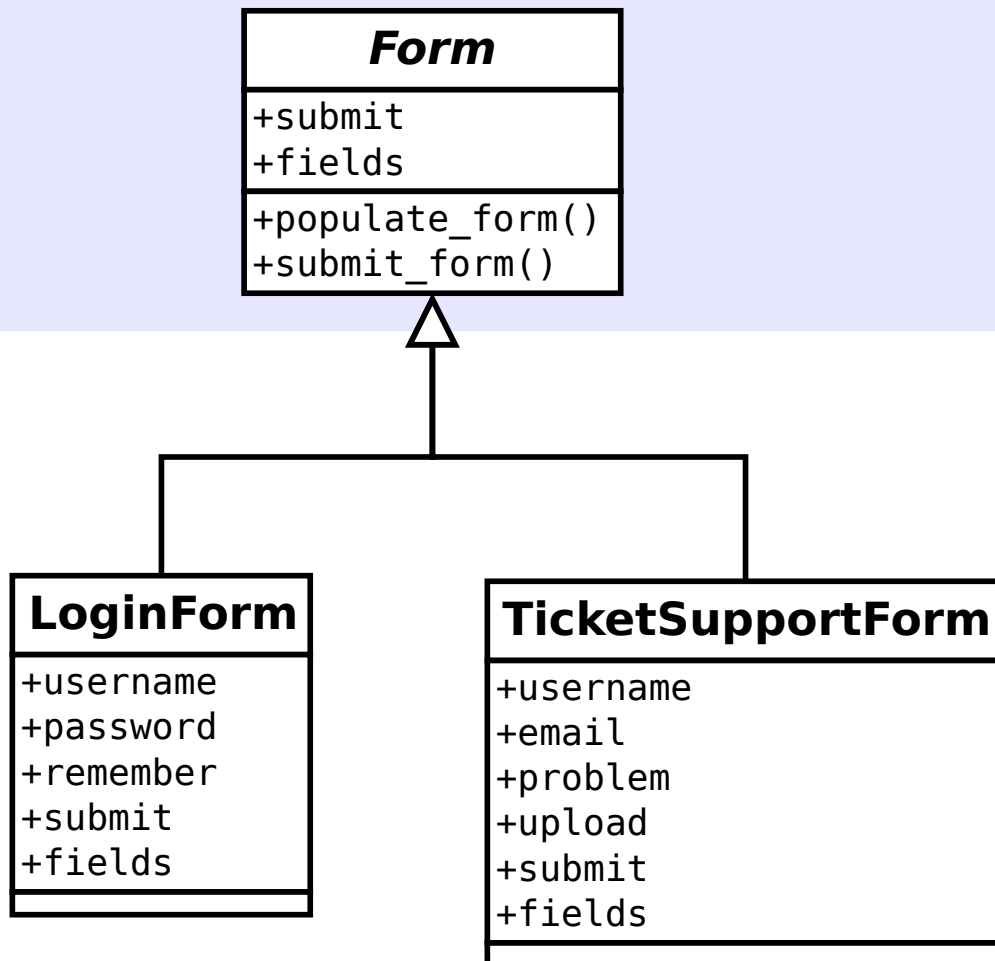
    def set_email(self, address):
        ...

    def submit_ticket(self, username, name, ...):
        ...
```



The screenshot shows a web form titled "Submit a Support Ticket" with a close button (X) in the top right corner. The form is organized into two columns. The left column contains three input fields: "Username:" (labeled "OPTIONAL"), "Name:" (labeled "REQUIRED"), and "E-mail:" (labeled "REQUIRED"). Below these is a dropdown menu with the text "Please answer the question: REQUIRED" and "Sharks live in..." followed by a selection box showing "- Select an answer -". The right column features a large text area for "Problem:" (labeled "REQUIRED") and a section for "Attach a screenshot:" (labeled "OPTIONAL") which includes a "Browse..." button, the text "No file selected.", and a list of supported file formats: ".jpg, .jpeg, .jpe, .bmp, .tif, .tiff, .png, .gif, .pub, .xml, .zip, .mp4, .pdf, .doc, .txt". A "Submit" button is located at the bottom right of the form.

# Proposed Interface: Form class



```
class Form(object):
```

```
...
```

```
def populate_form(self, data):
```

```
    for (k,v) in data:
```

```
        # find the widget in the object's
```

```
        # dictionary and set its value
```

```
        widget = getattr(self,k)
```

```
        widget.value = v
```

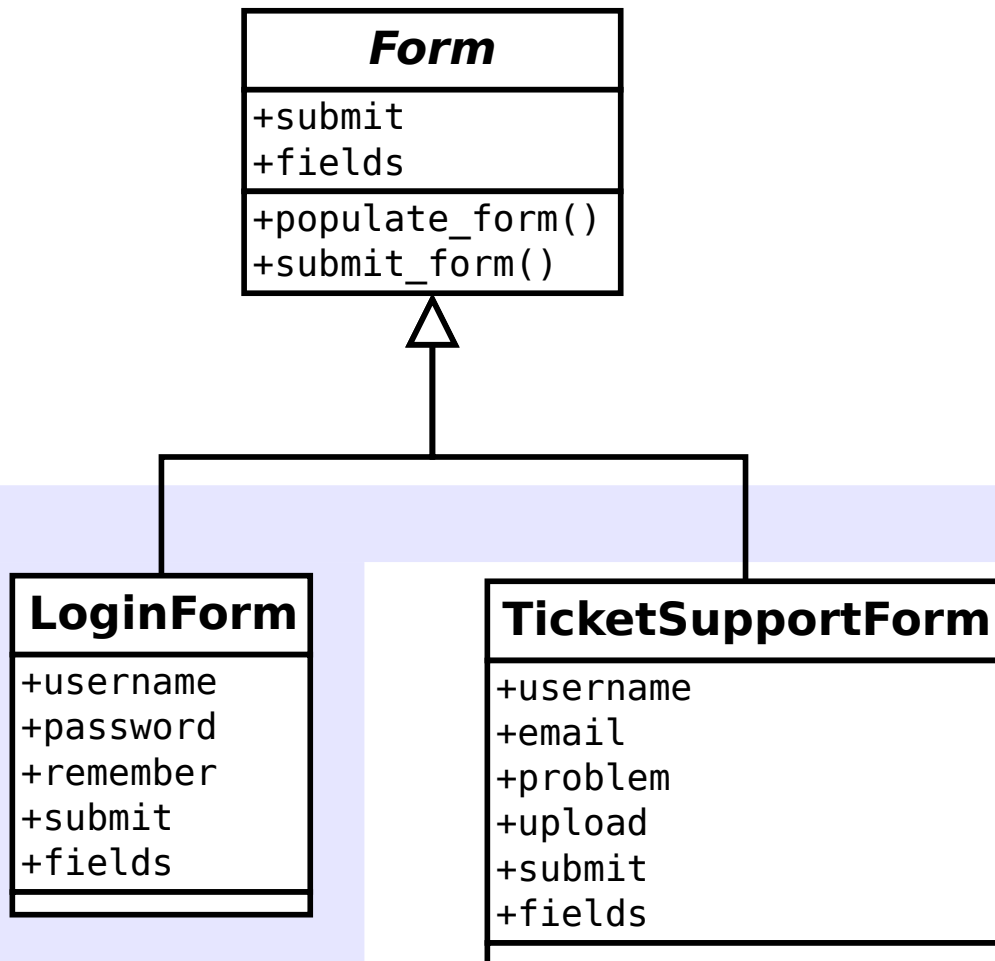
```
def submit_form(self,data={}):
```

```
    self.populate_form(data)
```

```
    result = self.submit.click()
```

```
    return result
```

# LoginForm subclasses Form

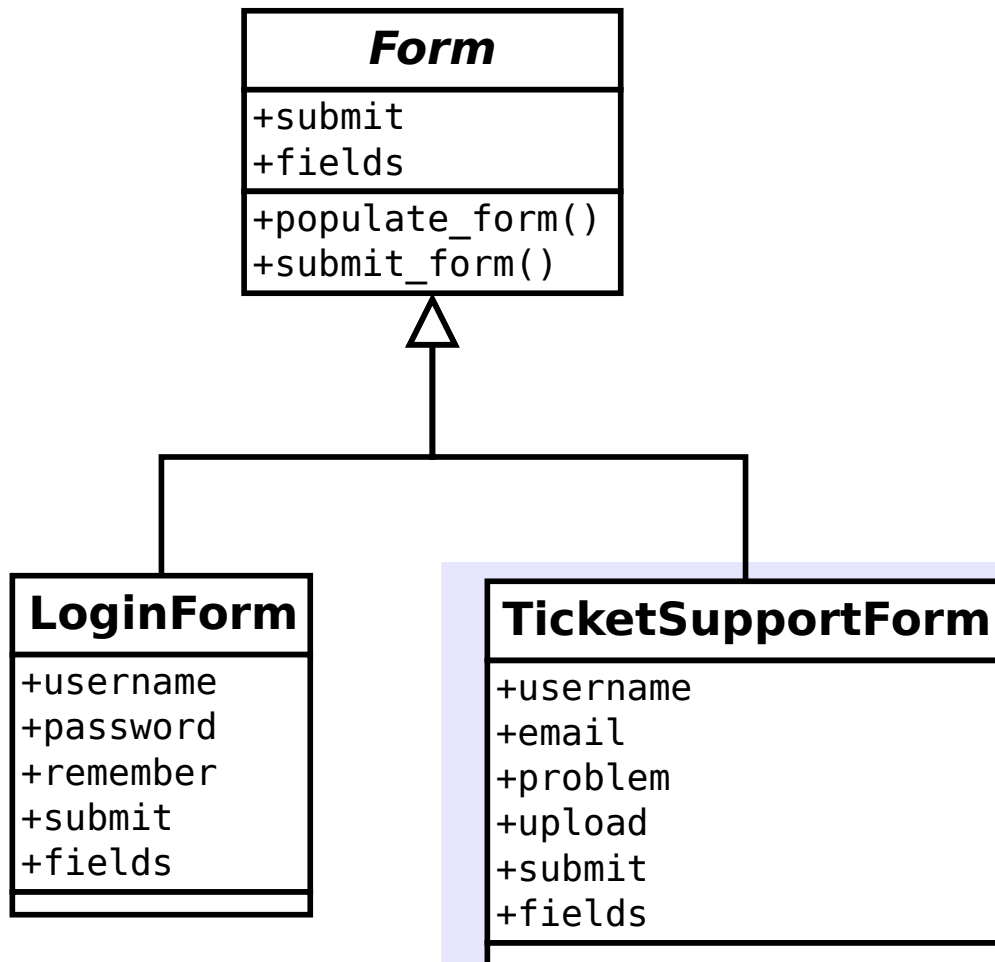


```
class LoginForm(Form):
```

```
    def __init__(self):
```

```
        self.username = Text('#username')
        self.password = Text('#password')
        self.remember = Checkbox('#rmbr')
        self.submit = Button('#submit')
```

# TicketSupportForm subclasses Form



```
class TicketSupportForm(Form):
```

```
    def __init__(self):
```

```
        self.name      = Text('#trName')
        self.email      = Text('#trEmail')
        self.problem     = Text('#trProblem')
        self.upload      = Text('#trUpload')
        self.submit      = Button('#submit')
```

# 3. Organize form data into dictionaries



## Knowledge Base

Find information on common issues.



## Ask the Community

Ask questions and find answers from other users.



## Wish List

Suggest a new site feature or improvement.



## Support Tickets

Check on status of your tickets.

Username:

OPTIONAL

Name:

REQUIRED

E-mail:

REQUIRED

Please answer the question:

REQUIRED

Eye, ankle or arm: which is part of the head?

Problem:

REQUIRED

Attach a screenshot:

OPTIONAL

Browse...

(.jpg, .jpeg, .jpe, .bmp, .tif, .tiff, .png, .gif)

Submit

```
po = TroubleReportForm()
```

```
po.submit_ticket(  
    'testuser',  
    'tu@hubzero.org',  
    'test problem',  
    'myscreenshot.png'  
)
```

```
po = TroubleReportForm()
```

```
data = {  
    'name'      : 'testuser',  
    'email'     : 'tu@hubzero.org',  
    'problem'   : 'test problem',  
    'upload'    : 'myscreenshot.png',  
}  
po.populate_form(data)  
  
po.submit_form()
```

# 3. Organize form data into dictionaries



## Knowledge Base

Find information on common issues.



## Ask the Community

Ask questions and find answers from other users.



## Wish List

Suggest a new site feature or improvement.



## Support Tickets

Check on status of your tickets.

Username:

OPTIONAL

Name:

REQUIRED

E-mail:

REQUIRED

Please answer the question:

REQUIRED

Eye, ankle or arm: which is part of the head?

Problem:

REQUIRED

Attach a screenshot:

OPTIONAL

(.jpg, .jpeg, .jpe, .bmp, .tif, .tiff, .png, .gif)

Submit

```
po = TroubleReportForm()
```

```
po.submit_ticket(  
    'testuser',  
    'tu@hubzero.org',  
    'test problem',  
    'myscreenshot.png'  
)
```

```
po = TroubleReportForm()
```

```
data = {  
    'name' : 'testuser',  
  
    'problem' : 'test problem',  
    'upload' : 'myscreenshot.png',  
}  
po.populate_form(data)  
  
po.submit_form()
```

# 3. Organize form data into dictionaries



## Knowledge Base

Find information on common issues.



## Ask the Community

Ask questions and find answers from other users.



## Wish List

Suggest a new site feature or improvement.



## Support Tickets

Check on status of your tickets.

Username:

OPTIONAL

Name:

REQUIRED

testuser

E-mail:

REQUIRED

Please answer the question:

REQUIRED

Eye, ankle or arm: which is part of the head?

Problem:

REQUIRED

test problem

Attach a screenshot:

OPTIONAL

Browse...

(.jpg, .jpeg, .jpe, .bmp, .tif, .tiff, .png, .gif)

Submit

```
po = TroubleReportForm()
```

```
po.submit_ticket(  
    'testuser',  
    'tu@hubzero.org',  
    'test problem',  
    'myscreenshot.png'  
)
```

```
po = TroubleReportForm()
```

```
data = {  
    'name' : 'testuser',  
    'problem' : 'test problem',  
}  
po.populate_form(data)  
po.submit_form()
```

# Extending the Web Form Pattern

Many types of Forms can extend the base class

- Simple Forms : fill-in order is ambiguous
- Ordered Forms : fill-in order matters
- Multi-Button Forms: forms with cancel, back, previous button
- Preview Forms : forms with a preview button



# Interacting with lists of items

The screenshot shows an eBay search results page for 'lenovo thinkpad x240'. The left sidebar contains filters for Categories, Brand (Lenovo), Product Line (ThinkPad), Screen Size (12-12.9 inches), Type (Notebook), Operating System, Memory, Hard Drive Capacity, Processor Speed, Condition (New), and Price. The main content area shows a single result for the 'Lenovo ThinkPad X240 - Windows Pro 64bit, i5, 500GB SSD, 8GB, w/ Warranty!!' for \$637.00. Below this, there are 'More Items related to "lenovo thinkpad x240"' including a 'Lenovo ThinkPad X61s' for \$20.50 and a 'Lenovo ThinkPad X61 Tablet' for \$50.00.

The screenshot shows an Amazon search results page for 'samsung 850 pro ssd'. The top navigation bar includes the Amazon logo, 'Your Amazon.com', 'Today's Deals', 'Gift Cards', 'Sell', and 'Help'. The search bar shows 'samsung 850 pro ssd' with a 'Go' button. Below the search bar, it says '1-16 of 28 results for "samsung 850 pro ssd"'. The results are categorized under 'Computers & Accessories' and 'Internal Solid State Drives'. Three product listings are visible: 'Samsung Electronics 850 Pro-Series 2.5" 256GB SATA III Internal Solid State Drive Single Unit Version MZ-7KE256BW...' for \$199.99, 'Samsung Electronics 850 Pro-Series 2.5" 512GB SATA III Internal Solid State Drive Single Unit Version MZ-7KE512BW...' for \$399.99, and 'Samsung Electronics 850 Pro-Series 2.5" 128GB SATA III Internal Solid State Drive Single Unit Version MZ-7KE128BW...' for \$129.99. Each listing includes a product image, title, price, and a 'Shop now' button.

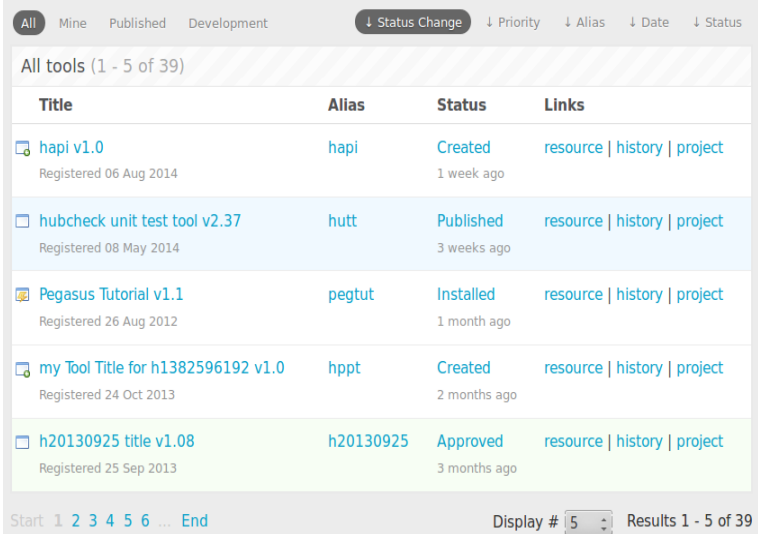
## Support: Tickets

The screenshot shows a support ticket management interface. At the top, there are buttons for 'STATS' and 'NEW TICKET'. Below this is a search bar with the text 'Search this query' and a 'Go' button. The main content area is divided into two sections: 'COMMON' and 'MINE'. The 'COMMON' section lists tickets with columns for 'Age', 'Status', 'Severity', 'Summary', 'Group', and 'Assignee'. The 'MINE' section lists tickets assigned to the user. The 'CUSTOM' section at the bottom allows filtering tickets by 'mine all', 'waiting', 'mine 2012', 'sdfs', 'all 2012', and 'lutgl'. The bottom of the page shows 'Start 1 2 3 4 5 6 ... End' and 'Display # 5 Results 1 - 5 of 713'.

The screenshot shows a tool management interface. At the top, there are buttons for 'All', 'Mine', 'Published', and 'Development'. Below this is a 'Status Change' button and a 'Priority' button. The main content area is a table with columns for 'Title', 'Alias', 'Status', and 'Links'. The table lists several tools, including 'hapi v1.0', 'hubcheck unit test tool v2.37', 'Pegasus Tutorial v1.1', 'my Tool Title for h1382596192 v1.0', and 'h20130925 title v1.08'. Each tool entry includes a title, alias, status, and links to 'resource', 'history', and 'project'. The bottom of the page shows 'Start 1 2 3 4 5 6 ... End' and 'Display # 5 Results 1 - 5 of 39'.

# Problems representing lists

1. Can't hard code locators
2. Don't want to pre-allocate items
3. Easy access to items
4. Sequential access to items
5. Searching for items



The screenshot shows a web interface for managing tools. At the top, there are tabs for 'All', 'Mine', 'Published', and 'Development'. A dropdown menu is set to 'Status Change'. Below the tabs, it says 'All tools (1 - 5 of 39)'. The main content is a table with four columns: Title, Alias, Status, and Links. The table lists five tools, each with a checkbox icon, a title, a registration date, an alias, a status, and a time ago indicator. The last row is highlighted in green.

Title	Alias	Status	Links
<input type="checkbox"/> hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39






# How are lists used?

1. Interact with list meta-data
2. Present enumerable data
3. Query list item value
4. Interact with list item

**Header counts**

**Column titles**

The screenshot shows a web application interface with a list of tools. At the top, there are tabs: 'All', 'Mine', 'Published', and 'Development'. Below these is a search bar and a dropdown menu for 'Status Change'. The main content area displays a table of tools. The table has four columns: 'Title', 'Alias', 'Status', and 'Links'. The first row is highlighted in blue. The second row is highlighted in light blue. The third row is highlighted in light green. The fourth row is highlighted in light green. The fifth row is highlighted in light green. The table is paginated, showing 'Results 1 - 5 of 39'.

Title	Alias	Status	Links
 hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	peg tut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39

# How are lists used?






1. Interact with list meta-data
2. **Present enumerable data**
3. Query list item value
4. Interact with list item

# Sequential

**1** 

**2** →

**3** 




All tools (1 - 5 of 39)				
Title	Alias	Status	Links	
 <b>hapi v1.0</b> Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>	
 <b>hubcheck unit test tool v2.37</b> Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>	
 <b>Pegasus Tutorial v1.1</b> Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>	
 <b>my Tool Title for h1382596192 v1.0</b> Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>	
 <b>h20130925 title v1.08</b> Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>	

# How are lists used?

1. Interact with list meta-data
2. Present enumerable data
3. Query list item values
4. Interact with list item

## Readable item detail

The diagram illustrates how a list of items is presented and how a specific item's details are expanded. A red box highlights the first item in a list, 'Pegasus Tutorial v1.1', which is also circled in red. A red arrow points from the text 'Readable item detail' to this item. Another red arrow points from the same text to the 'pegtut' alias and the 'Installed' status of the same item. A third red arrow points from the text to the 'Installed' status of the same item. The callout box shows the full details of the 'Pegasus Tutorial v1.1' item, including its icon, title, alias, status, and links to resource, history, and project.

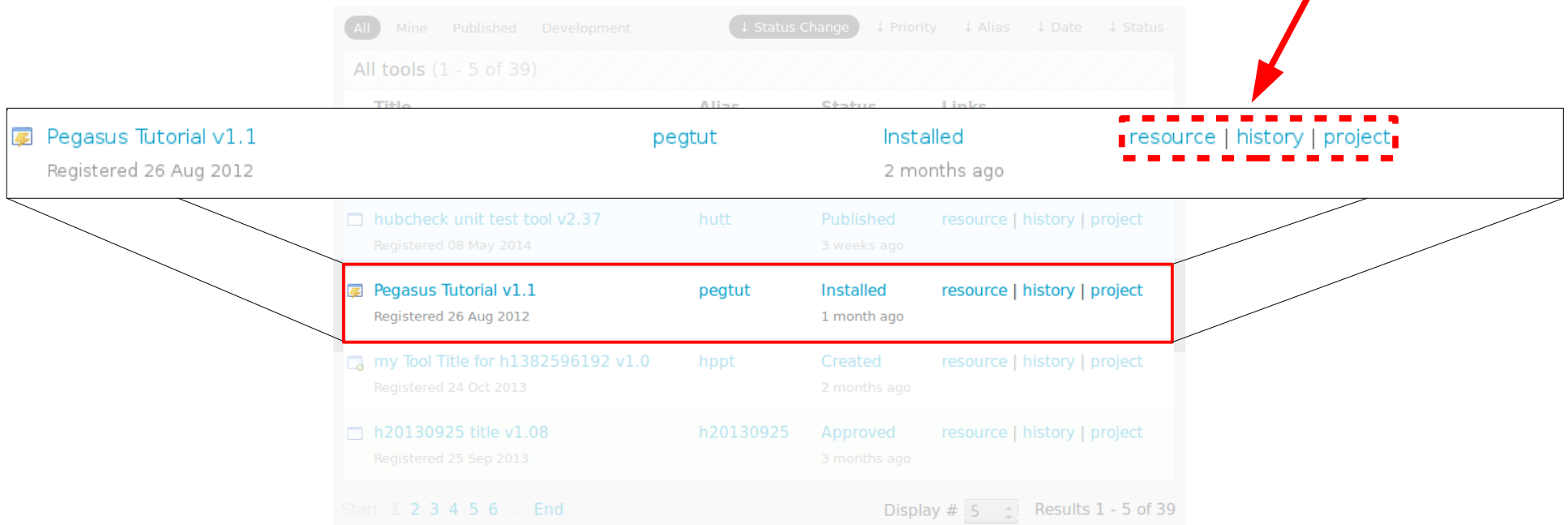
Title	Alias	Status	Links
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>



Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39

# How are lists used?

1. Interact with list meta-data
2. Present enumerable data
3. Query list item value
4. Interact with list item

Links to more detail

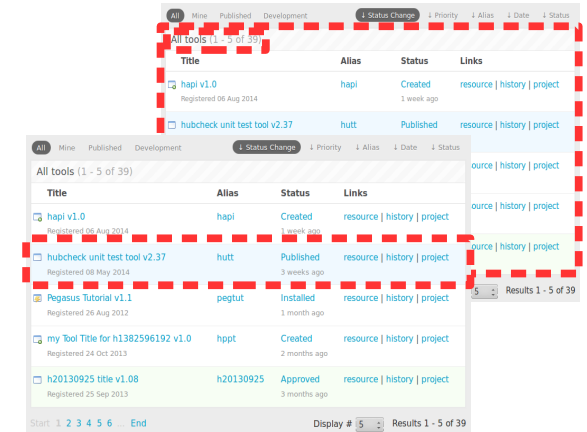


All Mine Published Development Status Change Priority Alias Date Status			
All tools (1 - 5 of 39)			
Title	Alias	Status	Links
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
Start 1 2 3 4 5 6 ... End			
Display # 5 Results 1 - 5 of 39			

# Proposed Interface

## ItemList Pattern

- Participants:
  - Container class
  - Item class
- Iterator Pattern to enumerate items
- Template Locators solve item count problem.
- Factory Method Pattern to create item objects on demand



All	Misc	Published	Development	Status Change	Priority	Alias	Date	Status
All tools (1 - 5 of 39)								
Title	Alias	Status	Links					
hapi v1.0 Registered 06 Aug 2014	hapi	Created	resource   history   project					
hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published	resource   history   project					
Pegasus Tutorial v1.1 Registered 26 Aug 2012	peg tut	Installed	resource   history   project					
my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created	resource   history   project					
h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved	resource   history   project					

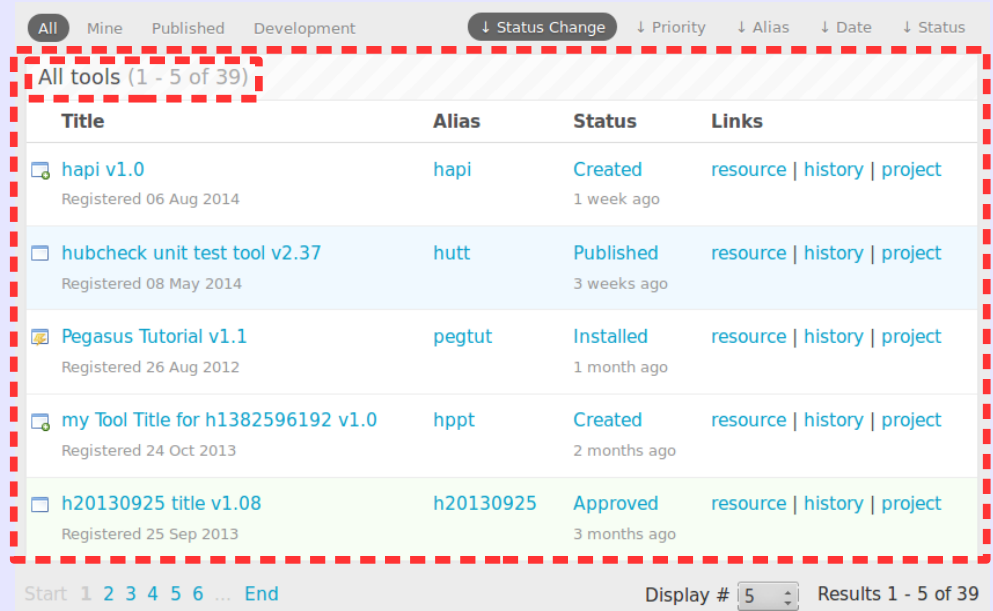
# ItemList Pattern Participants






## Container Class:

- Provides access to list meta-data  
num\_items()
- Can iterate through items  
next()
- Search for list items  
get\_item\_by\_position()  
get\_item\_by\_property()

## Item Class:

- Represents single item in list  
value()
- Can reference another item  
update\_item\_number()



All tools (1 - 5 of 39)			
Title	Alias	Status	Links
 hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39



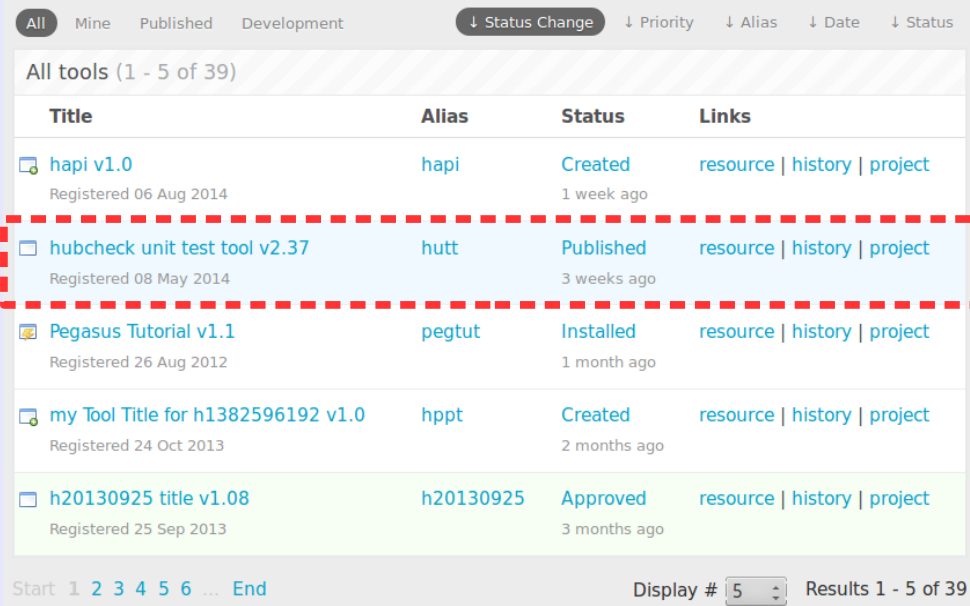
# ItemList Pattern Participants




## Container Class:

- Provides access to list meta-data  
num\_items()
- Can iterate through items  
next()
- Search for list items  
get\_item\_by\_position()  
get\_item\_by\_property()

## Item Class:

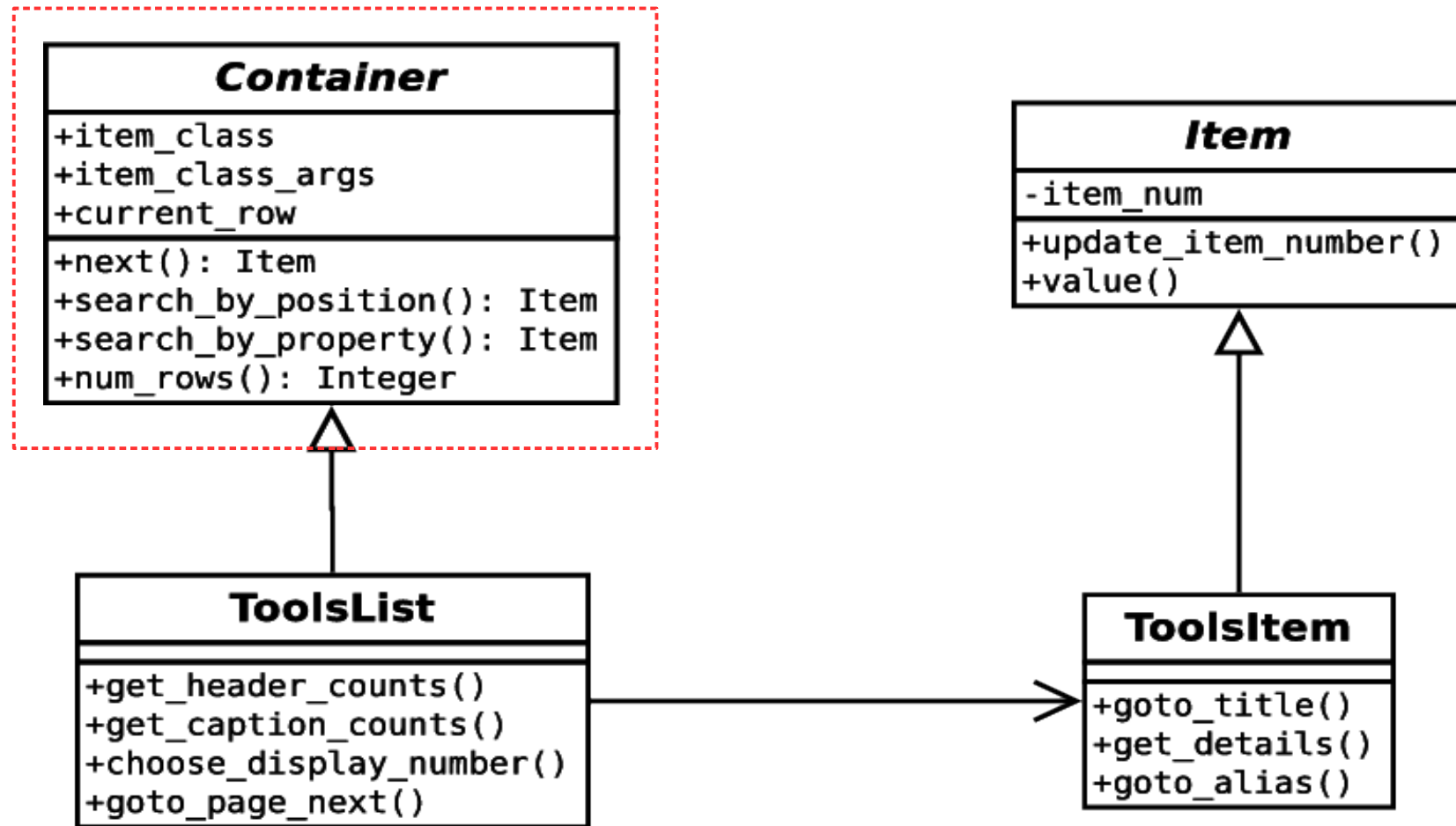
- Represents single item in list  
value()
- Can reference another item  
update\_item\_number()



All tools (1 - 5 of 39)			
Title	Alias	Status	Links
 hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39

# ItemList Pattern Participants



# Container Class Overview

```
class Container(object):
    def __init__(self, locatordict, item_class, *args):
        ...
    def num_items(self):
        ...
    def __iter__(self):
        ...
    def next(self):
        ...
    def get_item_by_position(self, item_number):
        ...
    def get_item_by_property(self, prop, val):
        ...
```

Access list  
meta-data |

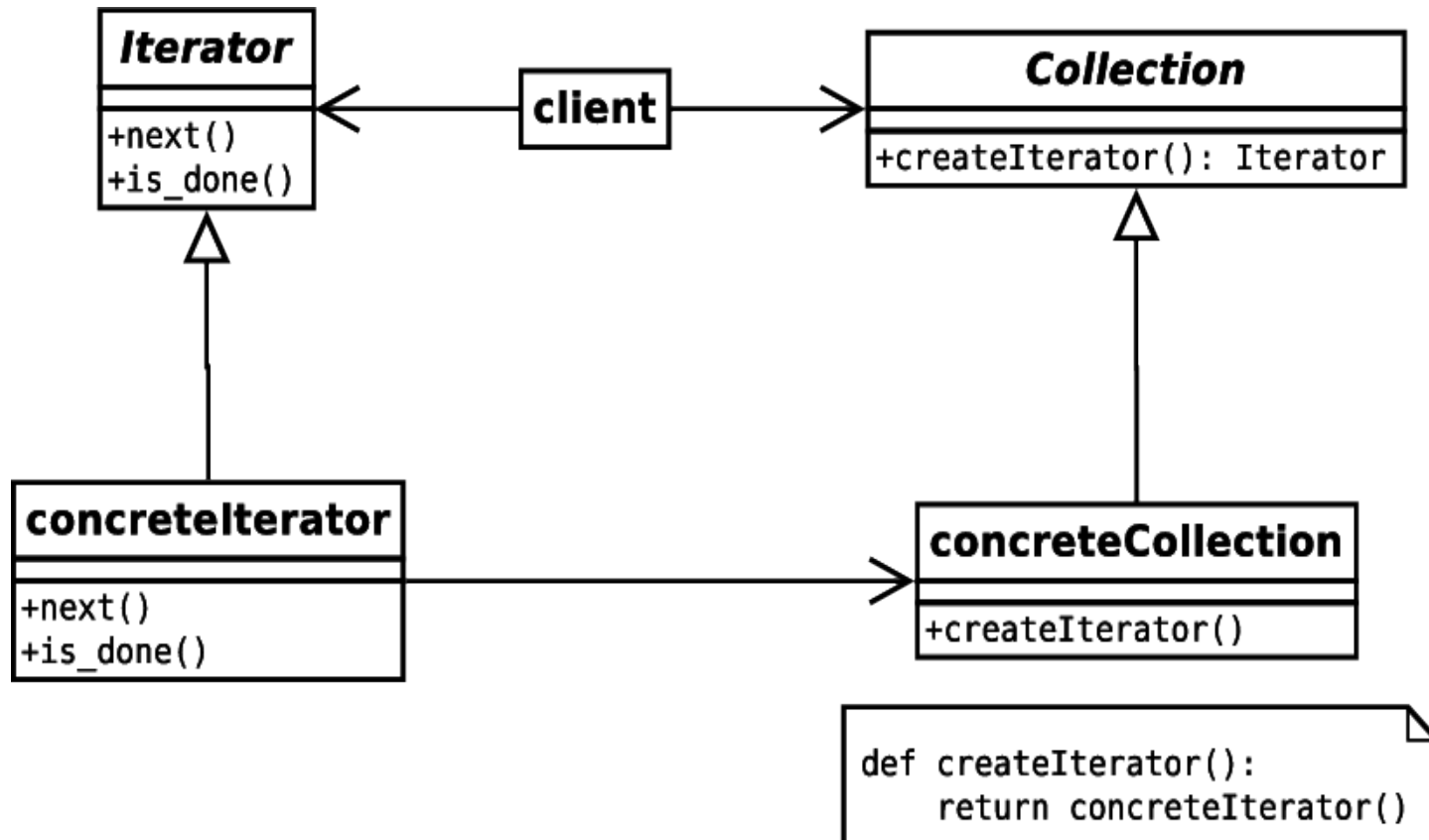
# Container Class Overview

```
class Container(object):
    def __init__(self, locator_dict, item_class, *args):
        ...
    def num_items(self):
        ...
    def __iter__(self):
        ...
    def next(self):
        ...
    def get_item_by_position(self, item_number):
        ...
    def get_item_by_property(self, prop, val):
        ...
```

**Iterate over  
items**

# Iterator Pattern

Sequentially access each element in a collection.



# Container Class Iterator

```
class Container(object):  
  
    def __iter__(self):  
        self.__current_item = 0  
        return self  
  
    def next(self):  
        ...  
        self.__current_item += 1  
  
        if self.__current_item >= self.__num_items:  
            # reset our counter, stop iterating  
            self.__current_item = 0  
            raise StopIteration  
  
        return self.get_item_by_position(self.__current_item)
```

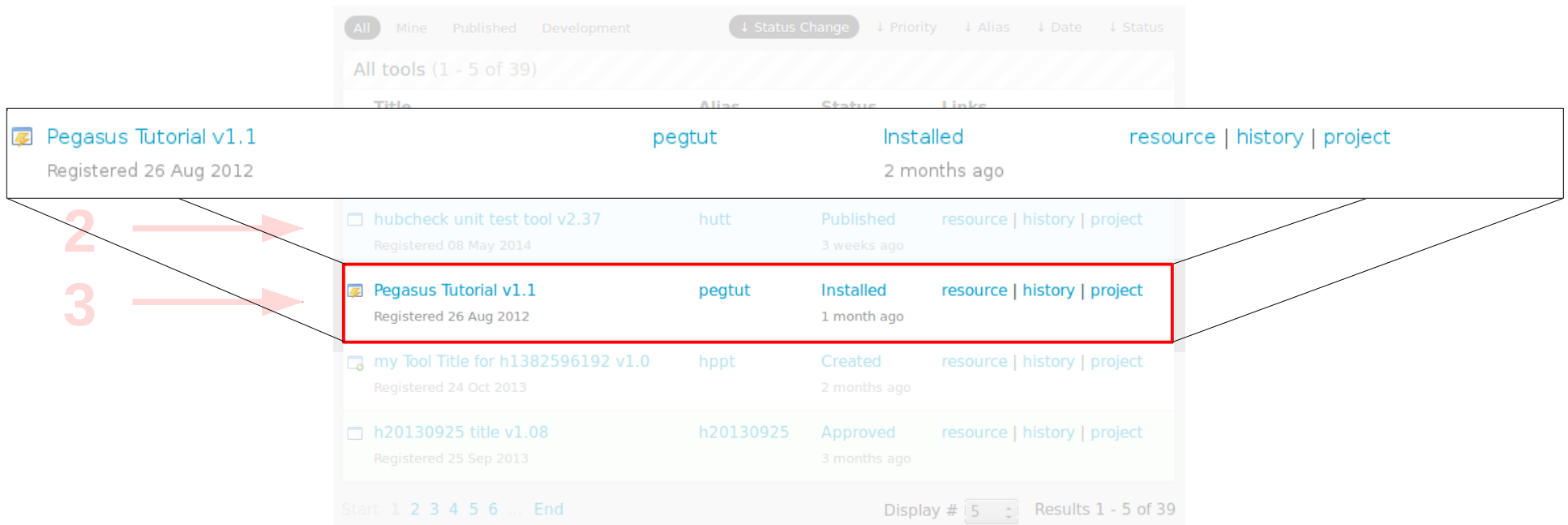
# Container Class Overview

```
class Container(object):
    def __init__(self, locator_dict, item_class, *args):
        ...
    def num_items(self):
        ...
    def __iter__(self):
        ...
    def next(self):
        ...
    def get_item_by_position(self, item_number):
        ...
    def get_item_by_property(self, prop, val):
        ...
```




**Search for  
list items**

# Searching for items: by position

```
def get_item_by_position(self, item_number):  
    ...  
    return Item(...)
```



The screenshot shows a web interface for managing tools. At the top, there are tabs for 'All', 'Mine', 'Published', and 'Development'. Below the tabs, there is a search bar and a list of tools. The table has columns for 'Title', 'Alias', 'Status', and 'Links'. The third item in the list is highlighted with a red box. Two red arrows point from the numbers '2' and '3' to the first and second items respectively.

Title	Alias	Status	Links
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39



# Searching for items: by property

```
def get_item_by_property(self, prop, val):  
    ...  
    return Item(...)
```

## Properties

Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End      Display # 5      Results 1 - 5 of 39

# Container Defines Item Class

ebay Shop by category

lenovo thinkpad x240

Related: lenovo thinkpad x230 lenovo thinkpad lenovo x230 ts130 lenovo thinkpad x1 carbon lenovo yoga 13 ke...

All Listings Auction Buy It Now Sort: Time: ending soonest View: [icon]

1 result for lenovo thinkpa... Follow this search

Lenovo ThinkPad 12-12.9 inches Notebook Clear All

**Lenovo Thinkpad X240 - Windows Pro 64bit, i5, 500GB SSD, 8GB, w/ Warranty!!**

12.5" Extended Battery, Backlit Keyboard, Touchpad

**\$637.00** 12h left (Today 9:23PM)

47 bids

More items related to "lenovo thinkpad x240"

**Lenovo ThinkPad X61s Windows 7 Premium Intel Core 2 Duo 2GB RAM 300GB HD WiFi**

**\$20.50** 6d 6h left (Monday, 2PM)

5 bids

**Lenovo ThinkPad X61 Tablet 7763 12.1" Notebook**

**\$50.00** 3d 9h left (Friday, 6PM)

1 bid

amazon Your Amazon.com Today's Deals Gift Cards Sell Help

Shop by Department Search All Departments samsung 850 pro ssd Go Hello, Sign in Your Account

1-16 of 28 results for "samsung 850 pro ssd"

Show results for

Computers & Accessories

Internal Solid State Drives

Electronics > Cell Phones & Accessories > Musical Instruments > See Fewer Departments

Refine by

Eligible for Free Shipping Free Shipping by Amazon

SSD Size

Brand

Internal Solid State Drive Size

Internal Solid State Drive Interface

**Studio 5.0 LTE: Blazing Speed 4G LTE, Unlocked**

Shop now

Related Searches: samsung 850 pro ssd, samsung 850, samsung 850 pro

**Samsung Electronics 850 Pro-Series 2.5" 256GB SATA III Internal Solid State Drive Single Unit Version MZ-7KE256BW...** by Samsung (Jul 21, 2014)

**\$199.99** **\$194.91** new (14 offers)

Only 1 left in stock - order soon.

More Buying Choices

FREE Shipping

Product Features

Product Details

Electronics: See all 27 items

**Samsung Electronics 850 Pro-Series 2.5" 512GB SATA III Internal Solid State Drive Single Unit Version MZ-7KE512BW...** by Samsung (Jul 21, 2014)

**\$399.99** **\$399.99** new (2 offers)

Get it by **Wednesday, Aug 20**

More Buying Choices

FREE Shipping

Product Features

Product Details

Electronics: See all 27 items

**Samsung Electronics 850 Pro-Series 2.5" 1TB SATA III Internal Solid State Drive Single Unit Version MZ-7KE128BW...** by Samsung (Jul 27, 2014)

**\$129.99** **\$129.99** new (20 offers)

Get it by **Wednesday, Aug 20**

More Buying Choices

FREE Shipping

Product Features

Product Details

Electronics: See all 27 items

## Support: Tickets

Search this query [Go]

COMMON

Open tickets 350

New tickets 19

Unassigned 18

Closed tickets 5542

MINE

Watch list 21

Reported by me 24

Assigned to me 5

CUSTOM

mine all 713

waiting 54

mine 2012 573

sdfs 26

all 2012 4771

lutgl 43

5999 Derrick Kearney (dkearney) 19 hours ago

In nees.org's debian 7 tool session containers, please install the following debian package: libgs10-dev

Pascal Meunier

5856 Derrick Kearney (dkearney) 2 weeks ago

In lemhub.org tool session containers, please check that submit and pegasus are installed. 2014-08-07 05:50:10,086: submit --local python /home/lemhub/hctest2/data/sessions/279/sayhi.py 2014-08-...

hubcheck testing

Steven Clark

5849 Derrick Kearney (dkearney) 2 weeks ago

on datacenterhub.org, please check that users are allowed to make webdav connections. derrick@gadabout:~\$ cadaver https://webdav.datacenterhub.org/webdav Could not connect to "webdav.datacenterhub..."

hubcheck testing

Pascal Meunier

5848 Derrick Kearney (dkearney) 2 weeks ago

In nees.org debian 7 tool session containers, please install the following debian package: libxmu6:i386 we are interested in the i386 version of the package for nanowhm, which is compiled with ...

Pascal Meunier

5846 Derrick Kearney (dkearney) 2 weeks ago

hapi

tool:hapi app:hapi

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 713

All Mine Published Development Status Change Priority Alias Date Status

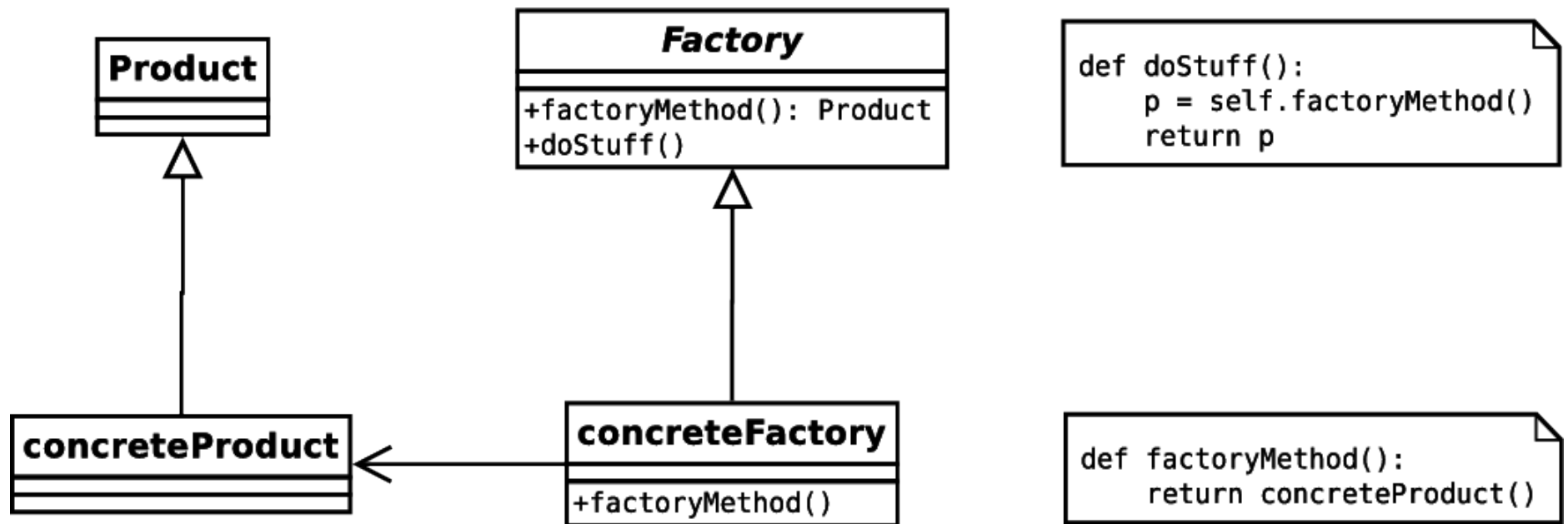
All tools (1 - 5 of 39)

Title	Alias	Status	Links
hapi v1.0 Registered 06 Aug 2014	hapi	Created	resource   history   project
hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published	resource   history   project
Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegut	Installed	resource   history   project
my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created	resource   history   project
h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved	resource   history   project

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39

# Factory Method Pattern

Define an interface for creating an object, but let subclasses define which class to instantiate.



# Searching for items

```
class Container(object):
    def __init__(self, locatordict):
        ...
        self.item_class = None
        self.item_class_args = None
        ...

    def get_item_by_position(self, item_number):
        ...
        result = self.item_class(
            *self.item_class_args,
            item_number=item_number )
        ...
        return result
```

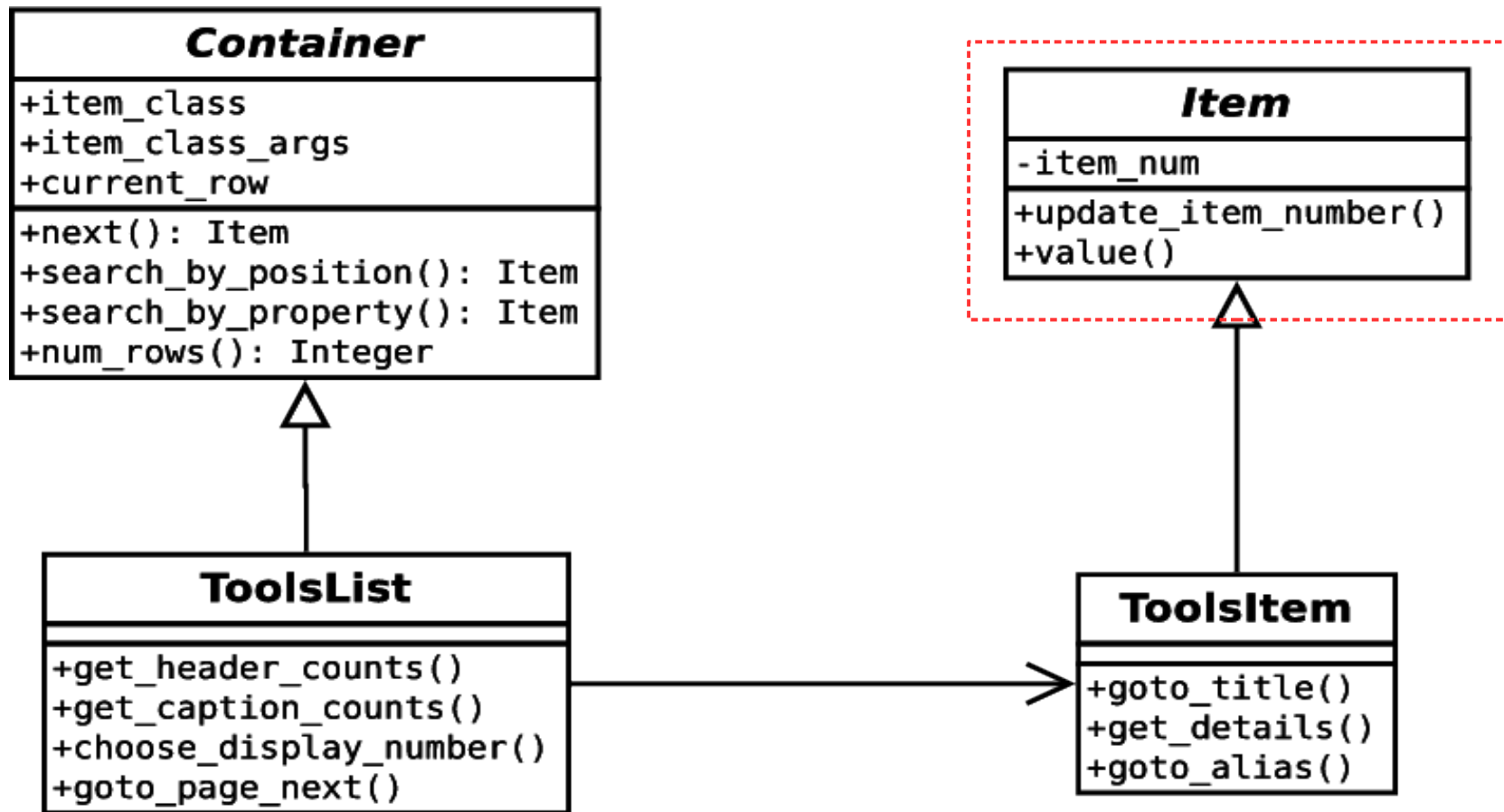
# ToolsList implements Container

```
class Container(object):
    def __init__(self, locatordict):
        ...
        self.item_class = None
        self.item_class_args = None
        ...
```

```
class ToolsList(Container):
    def __init__(self, locatordict):
        ...
        self.item_class = ToolsItem
        self.item_class_args = [{...}]
```

```
class ToolsItem(Item):
    def __init__(self, locator_templates):
        ...
```

# ItemList Pattern Participants



# Item Class Overview

```
class Item (object):
```

```
    def __init__ (self, locatordict, item_number):
```




```
        ...
```

```
    def update_item_number (self, item_number):
```

```
        ...
```

```
    def value (self):
```

```
        ...
```

All tools (1 - 5 of 39)			
Title	Alias	Status	Links
 hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1 Registered 26 Aug 2012	pegtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 my Tool Title for h1382596192 v1.0 Registered 24 Oct 2013	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
<input type="checkbox"/> h20130925 title v1.08 Registered 25 Sep 2013	h20130925	Approved 3 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Start 1 2 3 4 5 6 ... End Display # 5 Results 1 - 5 of 39

# Items represent a single item

```
class Item (object):
```

```
    def __init__(self, locatordict, item_number):  
        self.locators = locatordict  
        self.__item_number = item_number  
        ...
```

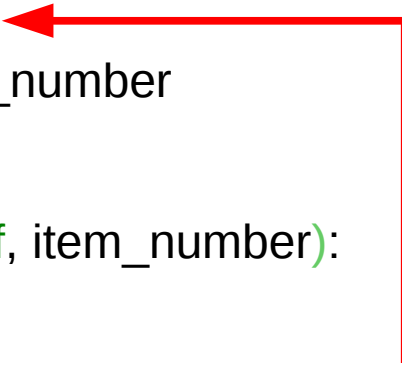


# Items can be updated

```
class Item (object):
```






```
    def __init__ (self, locatordict, item_number):  
        self.locators = locatordict  
        self.__item_number = item_number  
        ...
```

```
    def update_item_number (self, item_number):  
        ...
```



```
locators = {  
    'title'      : "tr:nth-of-type(1) .title",  
    'details'    : "tr:nth-of-type(1) .details",  
    'alias'      : "tr:nth-of-type(1) .alias",  
    'status'     : "tr:nth-of-type(1) .status",  
    'time'       : "tr:nth-of-type(1) .time",  
    'resource'   : "tr:nth-of-type(1) .page",  
    'history'    : "tr:nth-of-type(1) .history",  
    'wiki'       : "tr:nth-of-type(1) .wiki",  
}
```

# Interacting with ItemLists

All tools (1 - 5 of 39)			
Title	Alias	Status	Links
 <b>hapi v1.0</b> Registered 06 Aug 2014	<b>hapi</b>	<b>Created</b> 1 week ago	<b>resource</b>   <b>history</b>   <b>project</b>
 hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	resource   history   project
 Pegasus Tutorial v1.1	pgtut	Installed 1 month ago	resource   history   project
 hppt v1.0	hppt	Created months ago	resource   history   project
 h20130925		Approved	resource   history   project

tr:nth-of-type(1) .title

tr:nth-of-type(1) .alias






tr:nth-of-type(1) .status

tr:nth-of-type(1) .page

tr:nth-of-type(1) .history

tr:nth-of-type(1) .wiki

# Interacting with ItemLists

All	Mine	Published	Development	↓ Status Change	↓ Priority	↓ Alias	↓ Date	↓ Status
All tools (1 - 5 of 39)								
Title	Alias	Status	Links					
 hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	resource   history   project					
 hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	resource	history	project			
 Pegasus Tutorial v1.1	pgtut	Installed 1 month ago	resource   history   project					
 h20130925 v1.0	hppt	Created 2 months ago	resource   history   project					
 h20130925	h20130925	Approved	resource   history   project					

Display 4 of 39 results. Results 1 - 5 of 39

`tr:nth-of-type(2) .title`

`tr:nth-of-type(2) .alias`






`tr:nth-of-type(2) .status`

`tr:nth-of-type(2) .page`

`tr:nth-of-type(2) .history`

`tr:nth-of-type(2) .wiki`

# Interacting with ItemLists

All tools (1 - 5 of 39)			
Title	Alias	Status	Links
 hapi v1.0 Registered 06 Aug 2014	hapi	Created 1 week ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 hubcheck unit test tool v2.37 Registered 08 May 2014	hutt	Published 3 weeks ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 Pegasus Tutorial v1.1	pgtut	Installed 1 month ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 h20130925 v1.0	hppt	Created 2 months ago	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>
 h20130925	h20130925	Approved	<a href="#">resource</a>   <a href="#">history</a>   <a href="#">project</a>

Display 4 of 5 pages    Results 1 - 5 of 39

`tr:nth-of-type(n) .title`

`tr:nth-of-type(n) .alias`

`tr:nth-of-type(n) .status`

`tr:nth-of-type(n) .page`

`tr:nth-of-type(n) .history`

`tr:nth-of-type(n) .wiki`

# Template Locators

```
>>> locator = "tr:nth-of-type({item_num}) .title"  
>>>  
>>> locator.format(item_num=3)
```

*tr:nth-of-type(3) .title*

# Items can be updated

```
class Item (object):
```

```
def __init__ (self, locatorsdict, item_number):
```

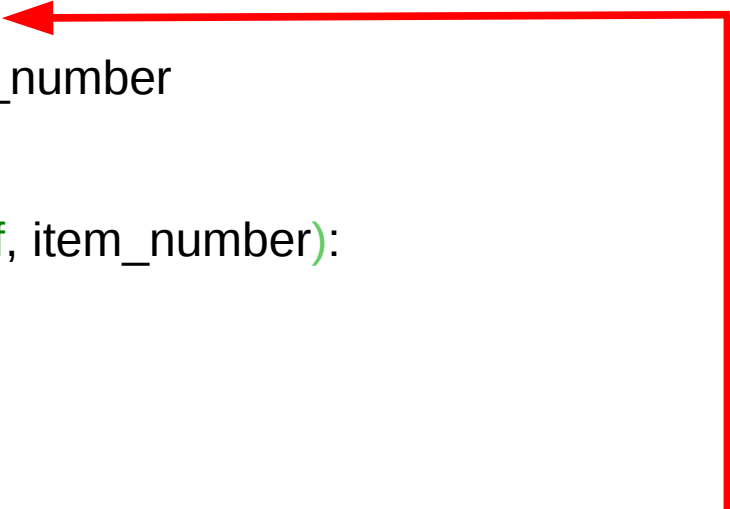
```
    self.locators = locatorsdict
```

```
    self.__item_number = item_number
```

```
    ...
```

```
def update_item_number (self, item_number):
```

```
    ...
```



```
locators = {  
    'title'      : "tr:nth-of-type(1) .title",  
    'details'   : "tr:nth-of-type(1) .details",  
    'alias'     : "tr:nth-of-type(1) .alias",  
    'status'    : "tr:nth-of-type(1) .status",  
    'time'      : "tr:nth-of-type(1) .time",  
    'resource'  : "tr:nth-of-type(1) .page",  
    'history'   : "tr:nth-of-type(1) .history",  
    'wiki'      : "tr:nth-of-type(1) .wiki",  
}
```

# Items can be updated

```
class Item (object):
```

```
    def __init__ (self, locator_templates, item_number):
```

```
        self.locators = {}
```

```
        self.locator_templates = locator_templates
```

```
        self.__item_number = item_number
```

```
        self.update_item_number(item_number)
```

```
        ...
```

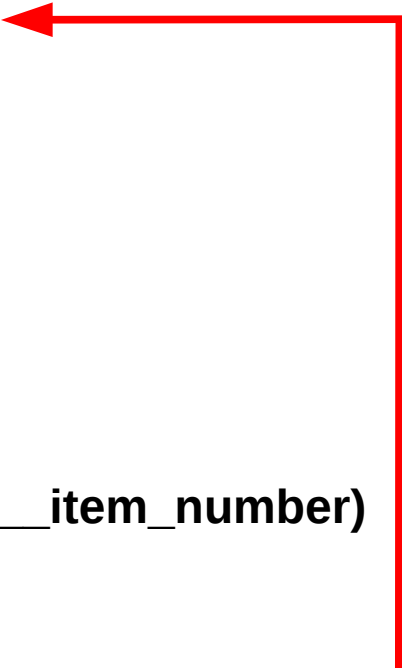
```
    def update_item_number (self, item_number):
```

```
        self.__item_number = item_number
```

```
        for k,v in self.locators_templates.items():
```

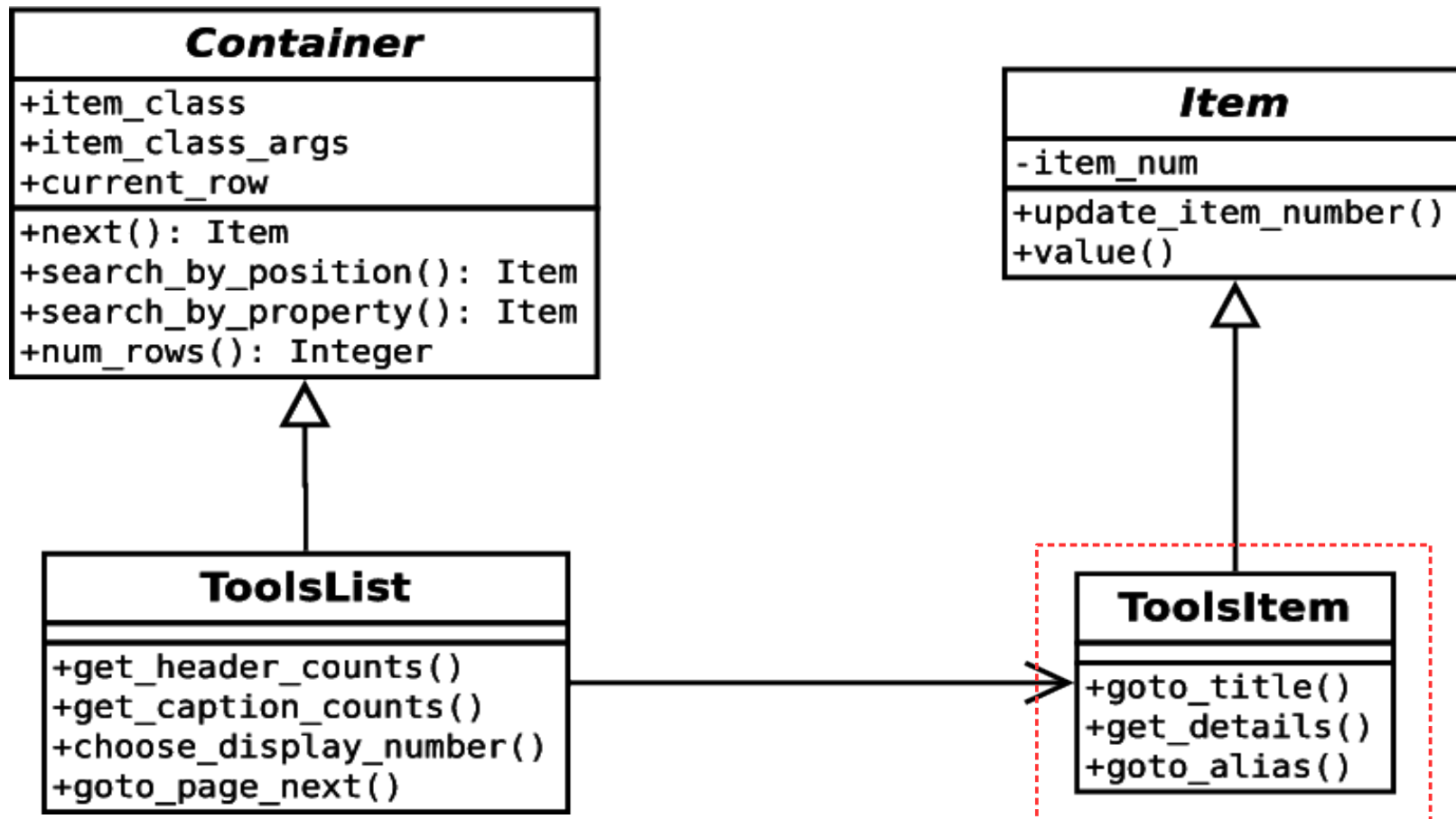
```
            self.locators[k] = v.format(item_num=self.__item_number)
```

```
        ...
```



```
locators = {  
    'title'      : "tr:nth-of-type({item_num}) .title",  
    'details'    : "tr:nth-of-type({item_num}) .details",  
    'alias'      : "tr:nth-of-type({item_num}) .alias",  
    ...  
}
```

# ItemList Pattern Participants





# Items have a value

```
class ToolItem (Item):
```

```
    def __init__ (self, locator_templates, item_number):
```

```
        ...
```

```
        self.title = Link(...)
```

```
        self.details = Text(...)
```

```
        self.alias = Link(...)
```

```
        ...
```

```
    def value(self):
```

```
        properties = {
```

```
            'title' : self.title.text(),
```

```
            'details' : self.details.value,
```

```
            'alias' : self.alias.text(),
```

```
            ...
```

```
        }
```

```
        return properties
```

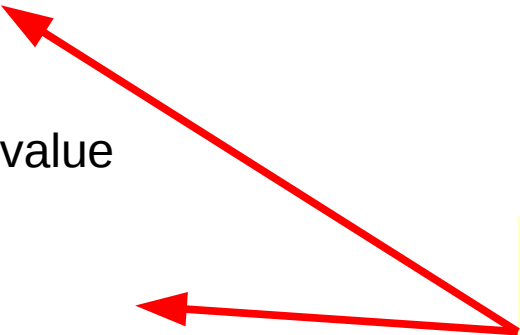
```
        ...
```



**value() method returns a dictionary of properties from the item.**

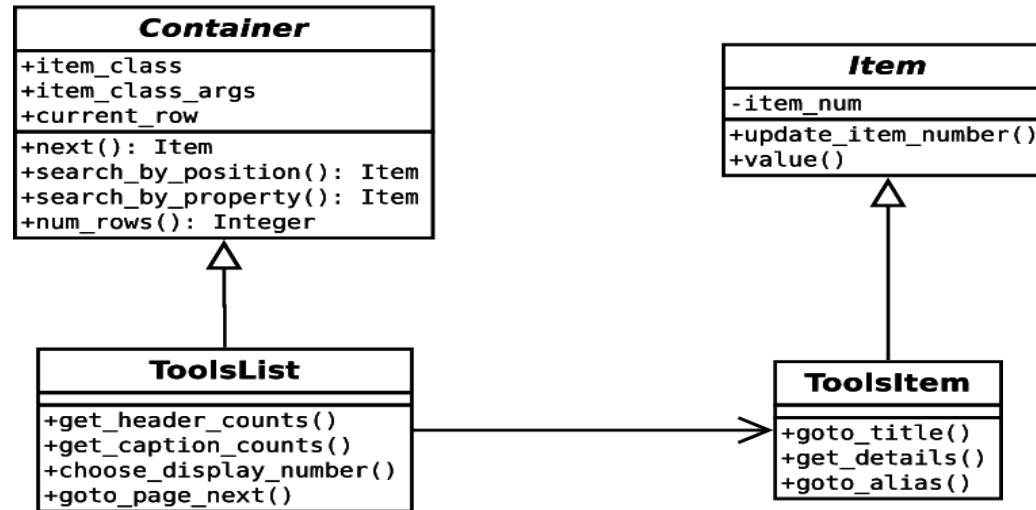
# Items can be interacted with

```
class ToolItem (object):  
    def __init__ (self, locator_templates, item_number):  
        ...  
        self.title = Link(...)  
        self.details = Text(...)  
        self.alias = Link(...)  
        ...  
  
    def goto_title(self):  
        self.title.click()  
  
    def get_details(self):  
        return self.details.value  
  
    def goto_alias(self):  
        self.alias.click()  
  
    ...
```



Web page services are implemented as methods of the page object.

# ItemList Pattern



## Problem

1. Can't hard code locators
2. Don't want to pre-allocate
3. Easy access to items
4. Sequential access to items
5. Searching for items

## Solution

1. Template Locators
2. Factory Method Pattern
3. Item objects are updatable
4. Iterator Pattern
5. Item objects have a value

# Interacting with Iframes

Abstract/Description:

This is abstract / description text

```
<label for="field-fulltxt">
```

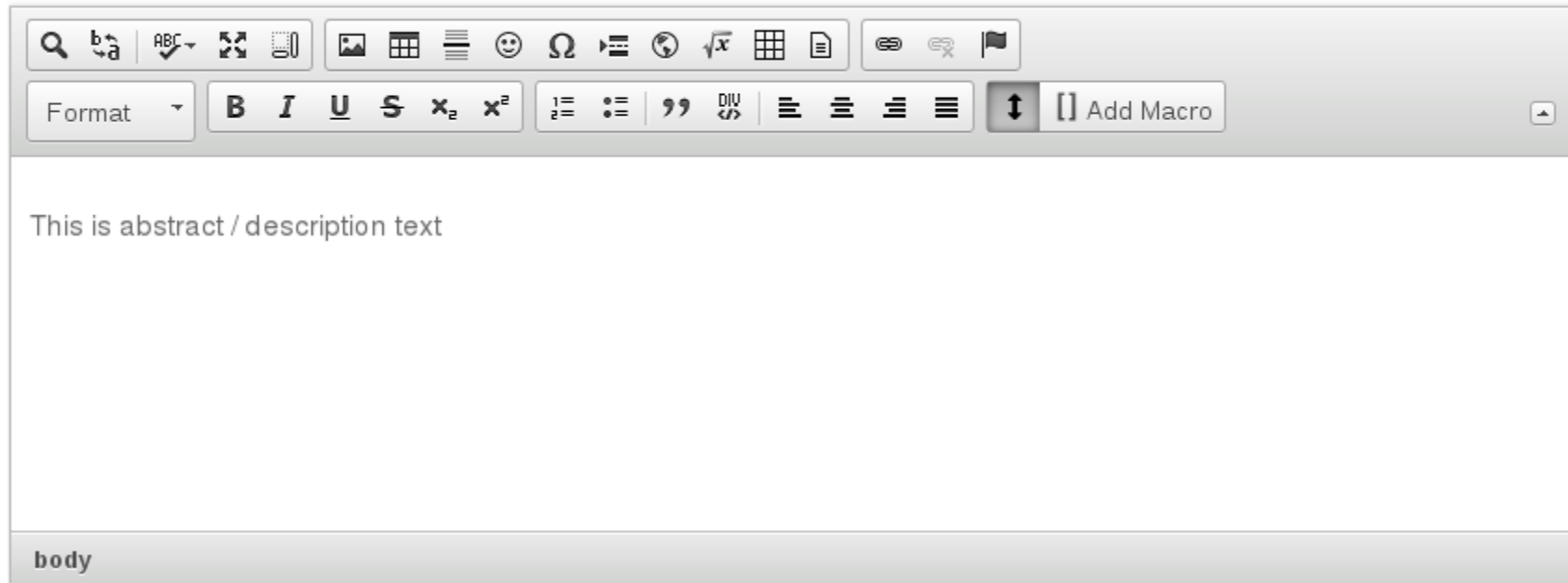
```
  Abstract/Description:
```

```
    <textarea id="field-fulltxt">This is abstract / description text</textarea>
```

```
</label>
```

# Interacting with Iframes

Abstract/Description:

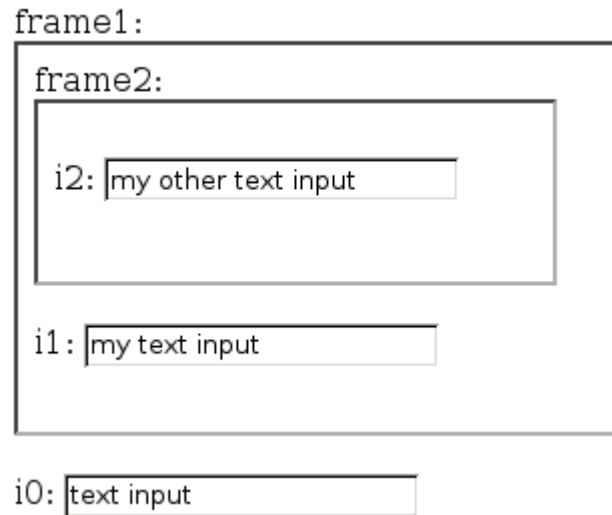


```
<iframe class="cke_wysiwig_frame">
  <html>
    <body class="ckeditor-body">
      <p>This is abstract / description text </p>
    </body>
  </html>
</iframe>
```

# Question:

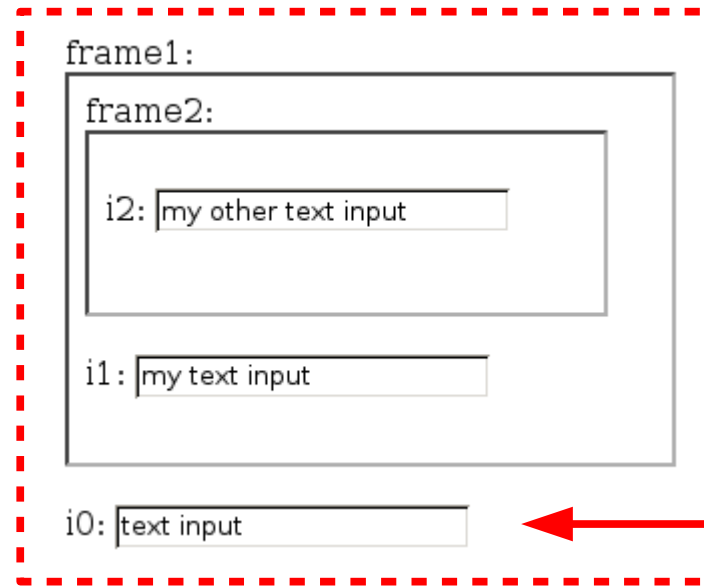
Do I need to write a new page object for a widget embedded in an <iframe>, if I already have a class that works except for entering and exiting the <iframe>?

# How do Iframes work



# How do Iframes work

Default Context

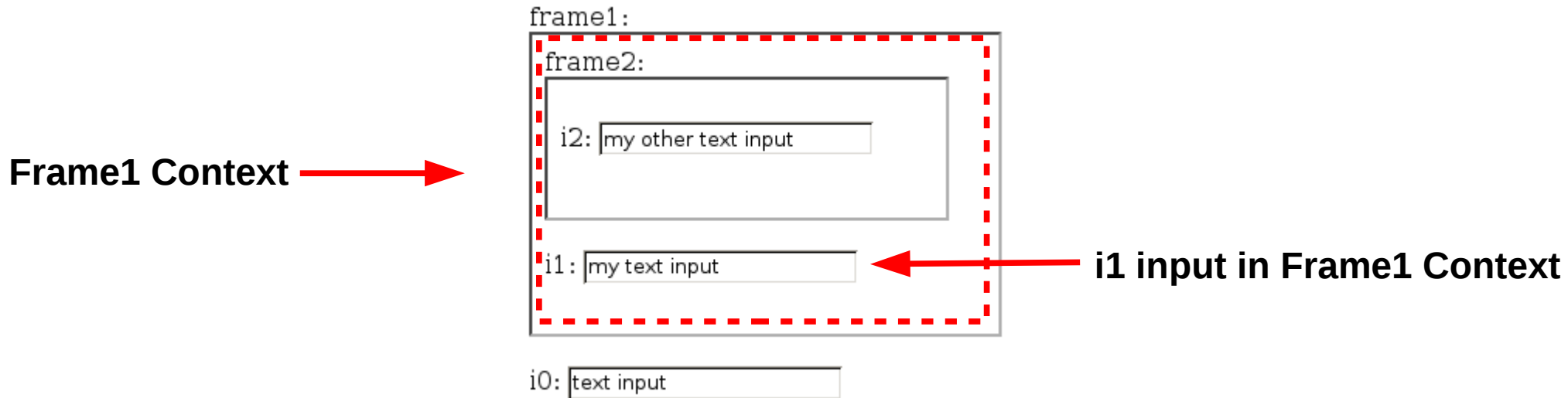


i0 input in Default Context

```
<html>
  <body>
    <label for="frame1">frame1: </label>
    <iframe id="frame1" src="inner_page.html"></iframe>
    <br/><br/>
    <label for="i0">i0: </label>
    <input type="text" id="i0" value="text input"></input>
  </body>
</html>
```

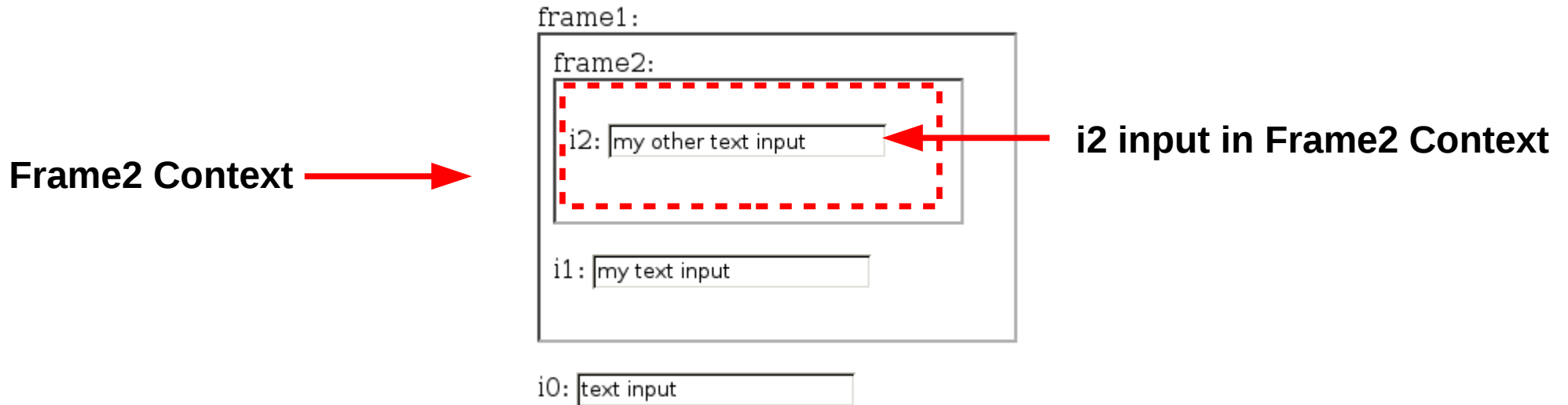


# How do Iframes work



```
<html>
  <body>
    <label for="frame2">frame2: </label>
    <iframe id="frame2" src="another_page.html"></iframe>
    <br/><br/>
    <label for="i1">i1: </label>
    <input type="text" id="i1" value="my text input"></input>
  </body>
</html>
```

# How do Iframes work



```
<html>
  <body>
    <label for="i2">i2: </label>
    <input type="text" id="i2" value="my other text input"></input>
  </body>
</html>
```

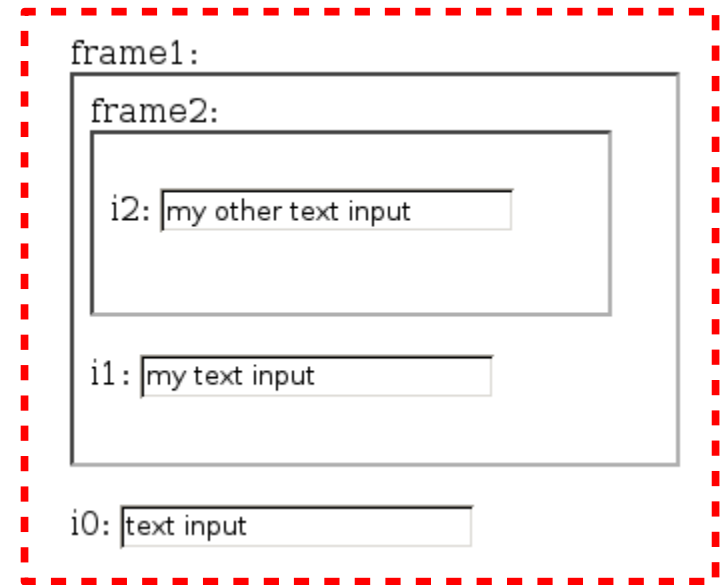
# Page Object for i0

```
class Text(BasePageWidget):
    def __init__(self, locator):
        ...

    # getter
    def value(self):
        e = self.find_element(self.locator)
        return e.get_attribute('value')

    # setter
    def value(self, text):
        e = self.find_element(self.locator)
        e.clear()
        e.send_keys(text)

    def append(self, text):
        e = self.find_element(self.locator)
        e.send_keys(text)
```



**Default Context**

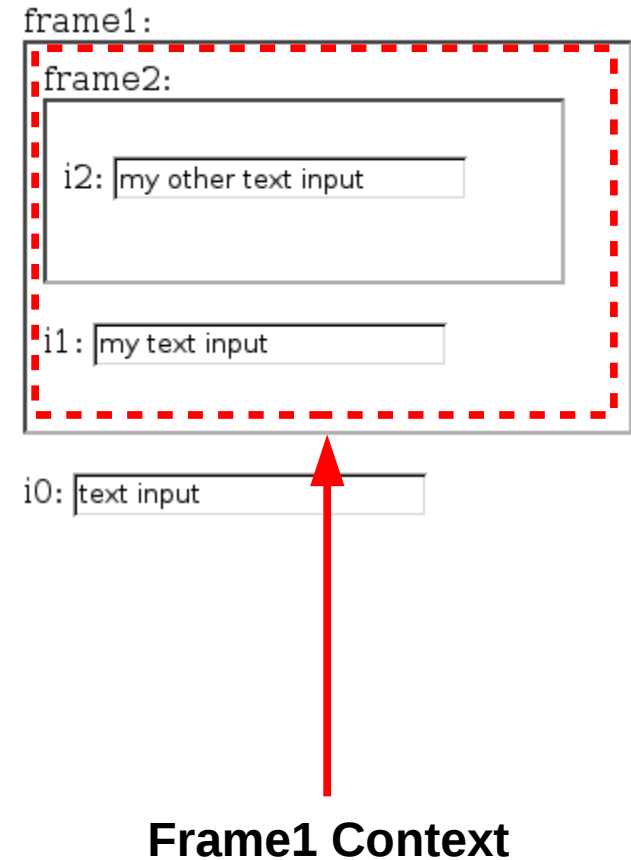
# Page Object for i1

```
class Text1Frame(BasePageWidget):
    def __init__(self, locator):
        ...

    # getter
    def value(self):
        ...

    # setter
    def value(self, text):
        frame = self.find_element('#frame1')
        self._browser.switch_to_frame(frame)
        e = self.find_element(self.locator)
        e.clear()
        e.send_keys(text)
        self._browser.switch_to_default_content()

    def append(self, text):
        ...
```



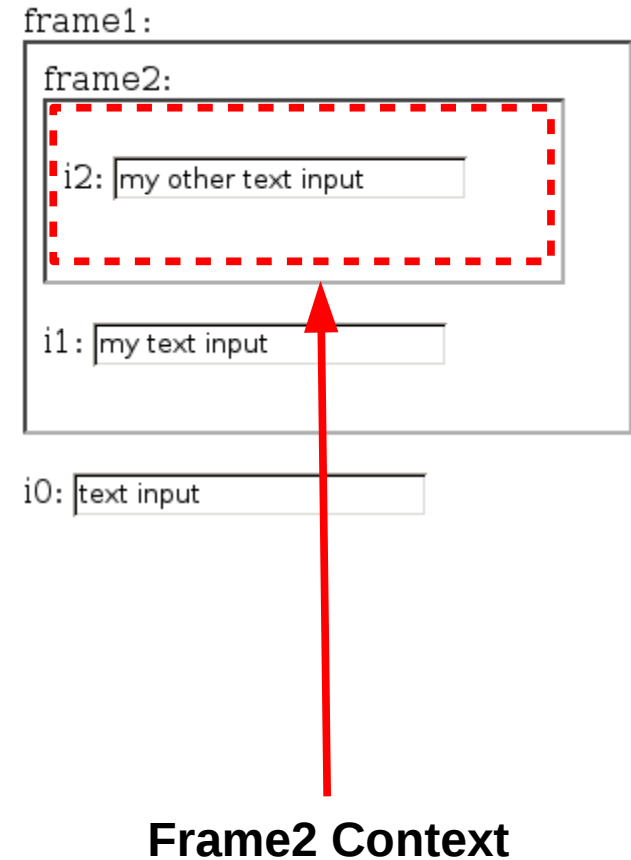
# Page Object for i2

```
class Text2Frame(BasePageWidget):
    def __init__(self, locator):
        ...

    # getter
    def value(self):
        ...

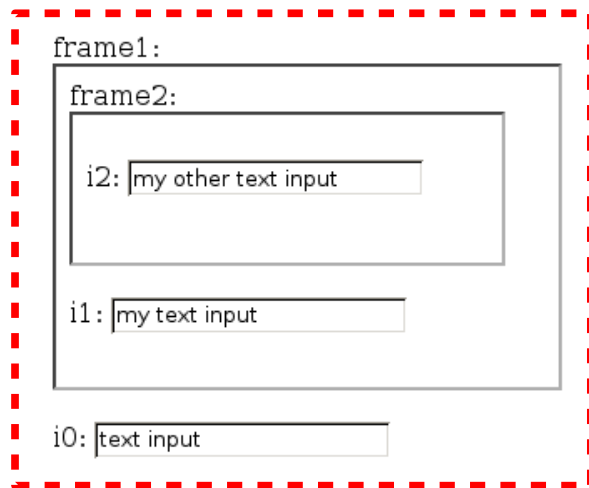
    # setter
    def value(self, text):
        frame1 = self.find_element('#frame1')
        self._browser.switch_to_frame(frame1)
        frame2 = self.find_element('#frame2')
        self._browser.switch_to_frame(frame2)
        e = self.find_element(self.locator)
        e.clear()
        e.send_keys(text)
        self._browser.switch_to_default_content()

    def append(self, text):
        ...
```

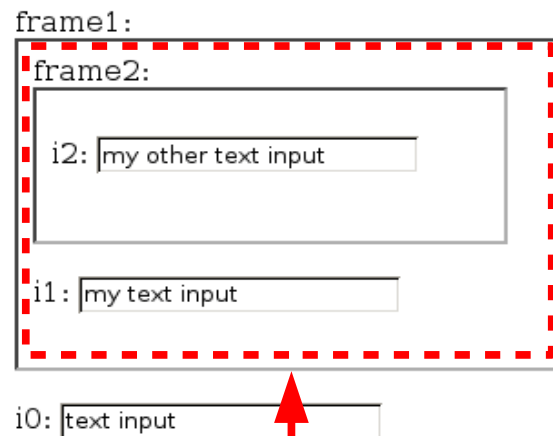


# Dealing with Iframes

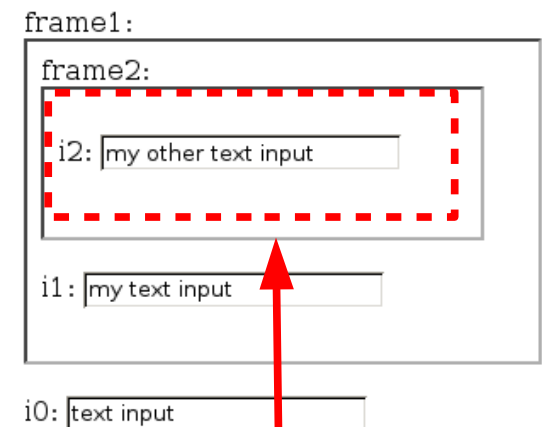
- Enter / Exit frames to interact with elements
- Must update all methods of page object.
- Create new page objects for Iframes?!?



**Default Context**



**Frame1 Context**

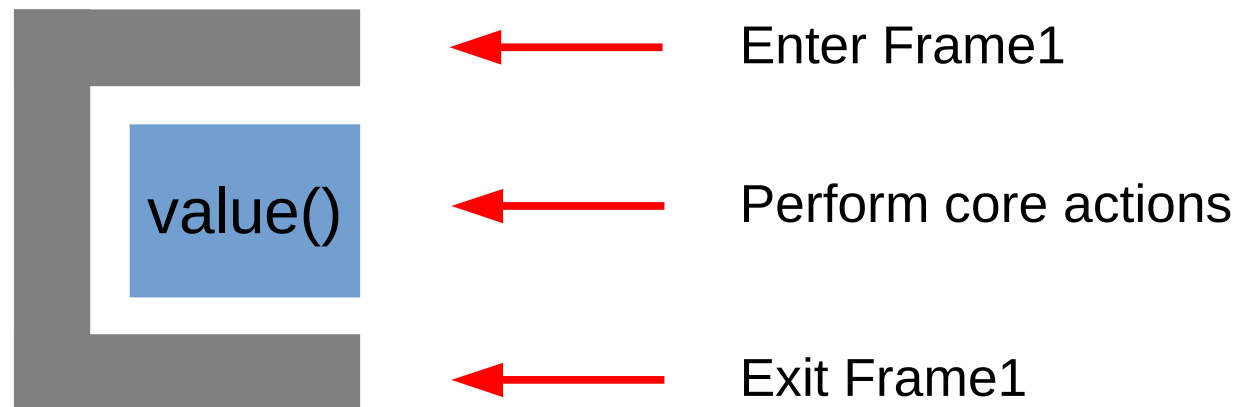


**Frame2 Context**

# Page object's value: No Iframes

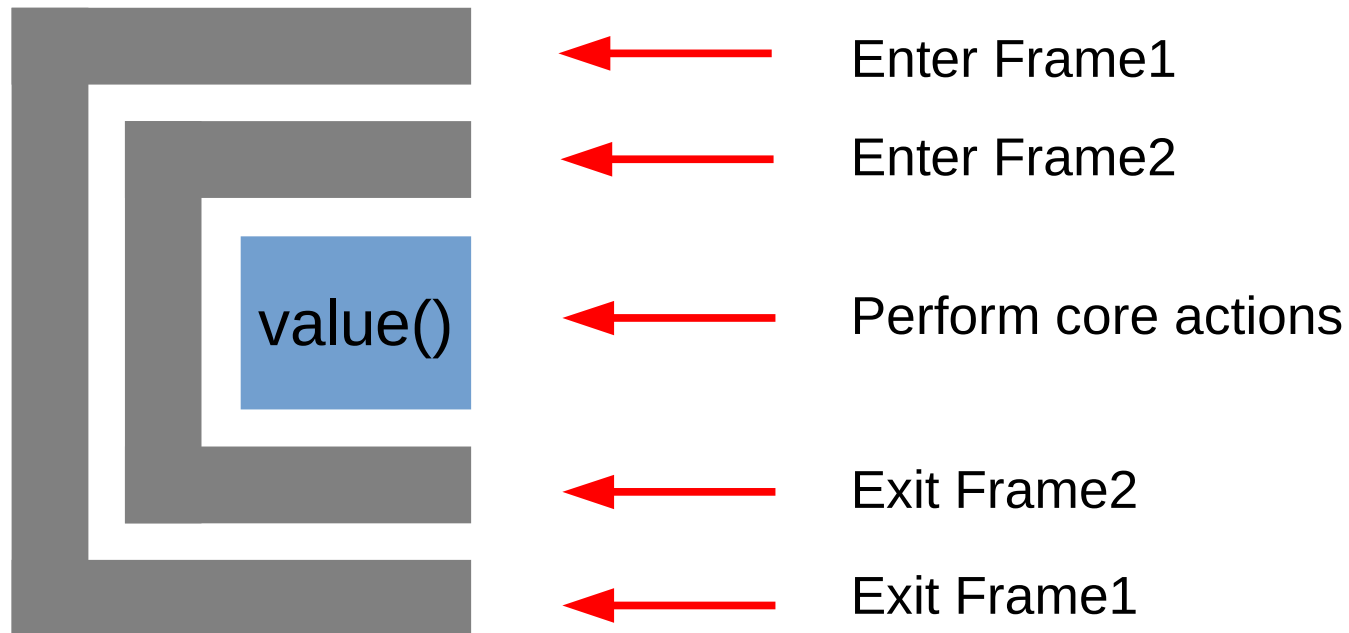


# Page object's value: One Iframe



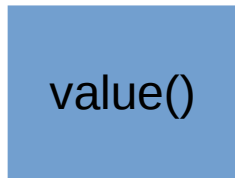


# Page object's value: Two Iframes

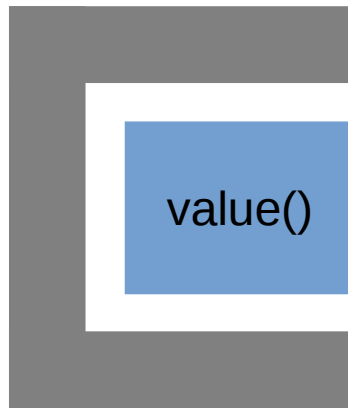


# IframeWrap Design Pattern

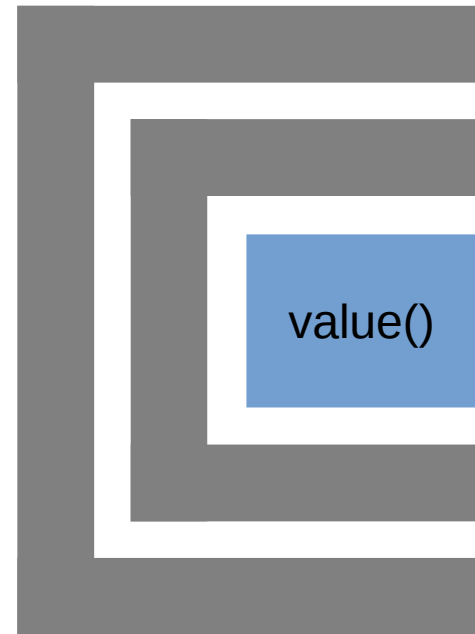
Use Decorator Pattern to wrap attributes of a page object with Enter / Exit iframe calls.



**0 Iframes**



**1 Iframe**



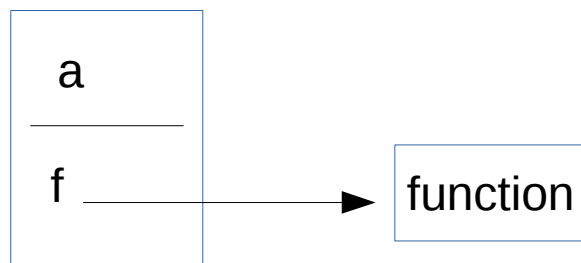
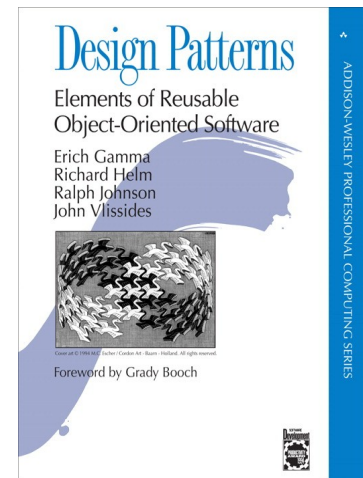
**2 Iframes**

# Decorator Pattern

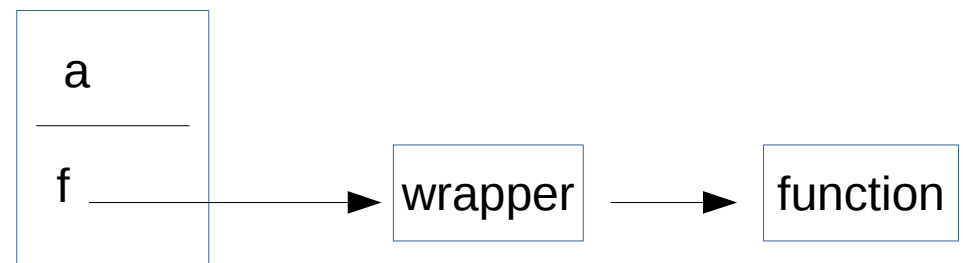
Attach additional responsibilities to an object dynamically and transparently.

```
class A(object):  
  
    def f(self):  
        do_some_stuff()
```

```
a = A()
```



**Before Decoration**

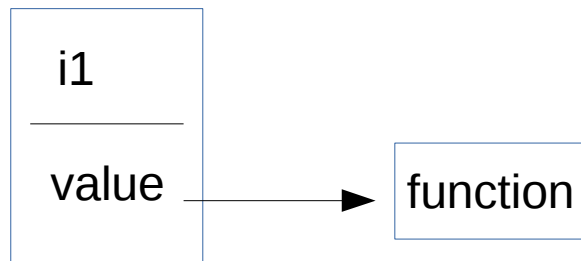
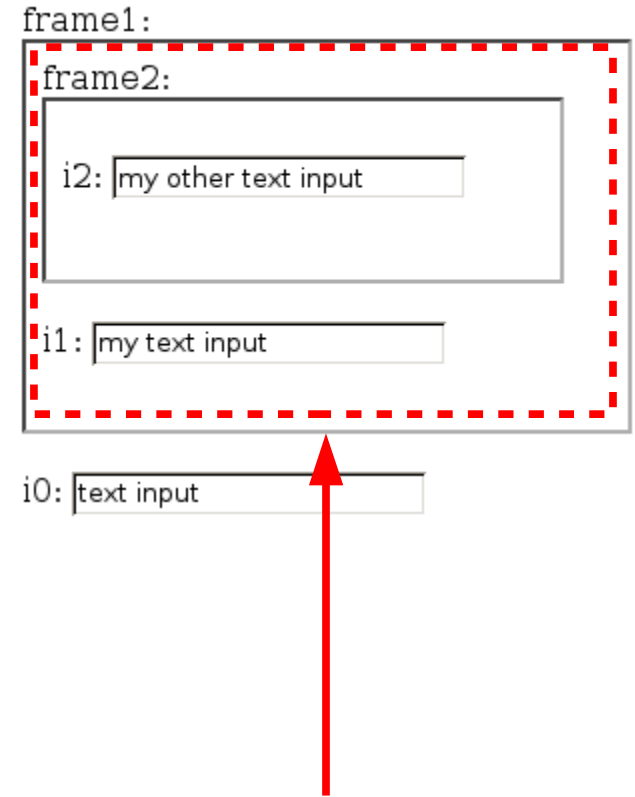


**After Decoration**

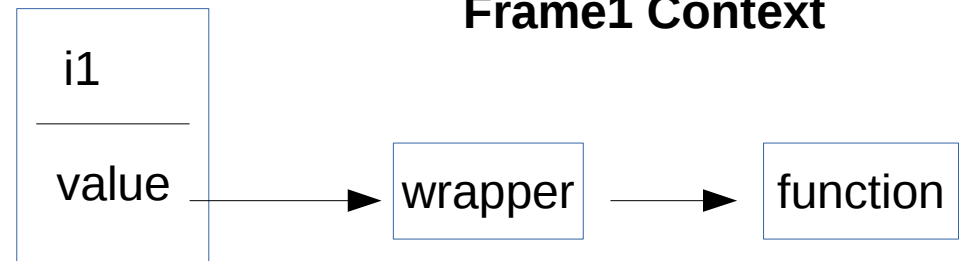
# Applying the Decorator Pattern

```
class Text(BasePageWidget):  
  
    # setter  
    def value(self, text):  
        e = self.find_element(self.locator)  
        e.clear()  
        e.send_keys(text)
```

```
i1 = Text('#i1')  
i1.value('new i1 text')
```



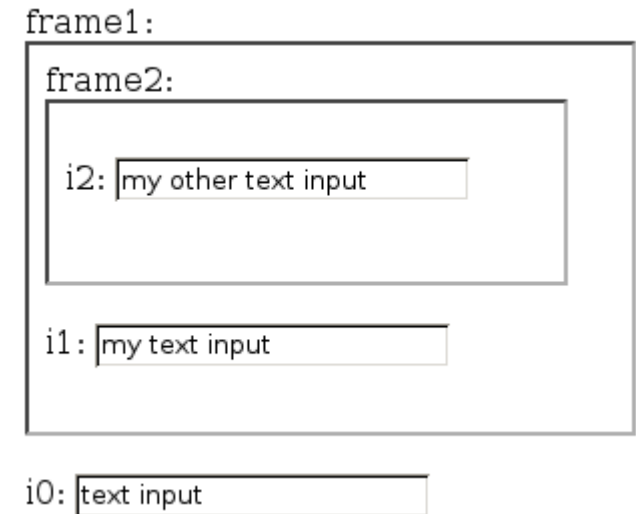
**Before Decoration**



**After Decoration**

# Create decorated page objects

```
class FramedInputs(BasePageObject):  
    def __init__(self):  
  
        self.i0 = Text('#i0')  
  
        self.i1 = IFrameWrap( Text('#i1'), ['#frame1'] )  
  
        self.i2 = IFrameWrap( Text('#i2'), ['#frame2', '#frame1'] )
```

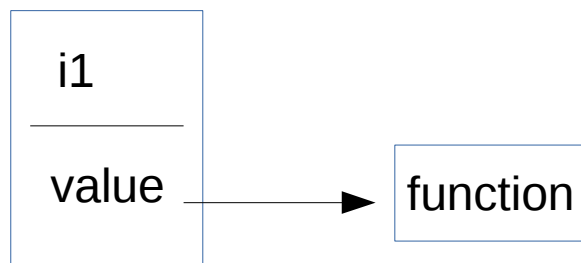


HTML element's  
page object

List of frames  
to traverse

# Find callable attributes

```
class IframeTracker(object):  
    ...  
    def wrap_callable_attributes(self,o):  
        for attr, item in o.__class__.__dict__.items():  
            if callable(item):  
                item = getattr(o,attr)  
                setattr(o,attr,self.wrap_attribute(item))
```



**Before Decoration**



**After Decoration**

# Wrap attributes with iframe calls

```
class IframeTracker(object):
```

```
...
```

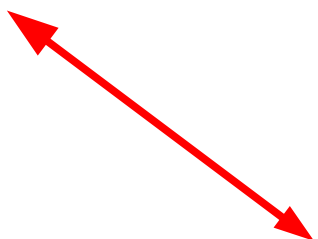
```
def wrap_callable_attributes(self,o):  
    for attr, item in o.__class__.__dict__.items():  
        if callable(item):  
            item = getattr(o,attr)  
            setattr(o,attr,self.wrap_attribute(item))
```

```
def wrap_attribute(self,item):  
    def wrapper(*args, **kwargs):
```

```
        ...
```

```
        switched = self._switch_to_iframe_context(final_framelevel)  
        result = item(*args, **kwargs)  
        self._switch_to_iframe_context(initial_framelevel)
```

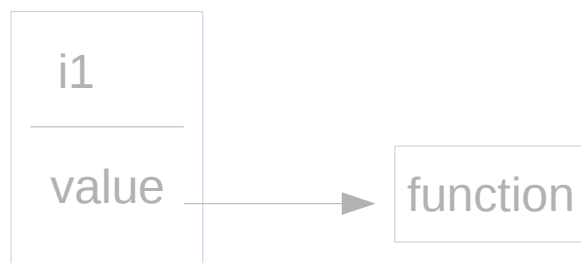
```
    return result  
    return wrapper
```



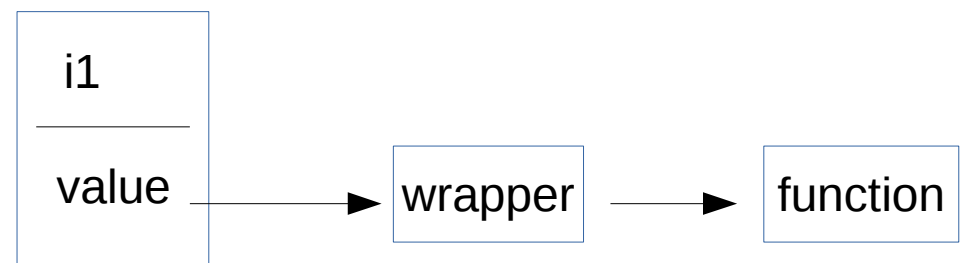
```
        frame = self.find_element('#frame1')  
        self._browser.switch_to_frame(frame)  
        e = self.find_element(self.locator)  
        e.clear()  
        e.send_keys(text)  
        self._browser.switch_to_default_content()
```

# Store the wrapped attribute

```
class IframeTracker(object):  
    ...  
    def wrap_callable_attributes(self,o):  
        for attr, item in o.__class__.__dict__.items():  
            if callable(item):  
                item = getattr(o,attr)  
                setattr(o,attr,self.wrap_attribute(item))
```



Before Decoration



After Decoration



# Using an IframeWrap'd page object

```
class FramedInputs(BasePageObject):  
    def __init__(self):  
        self.i0 = Text('#i0')  
        self.i1 = IframeWrap( Text('#i1'), ['#frame1'] )  
        self.i2 = IframeWrap( Text('#i2'), ['#frame2', '#frame1'] )
```

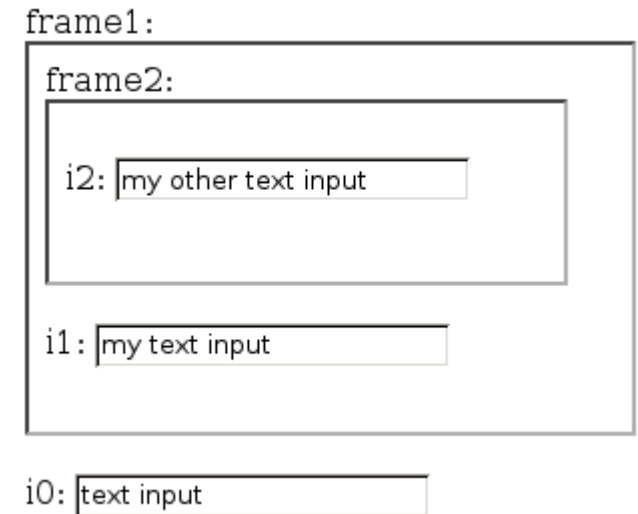
```
po = FramedInputs()
```

*# print out the current text in the widgets*

```
print "i0.value = %s" % (po.i0.value)  
print "i1.value = %s" % (po.i1.value)  
print "i2.value = %s" % (po.i2.value)
```

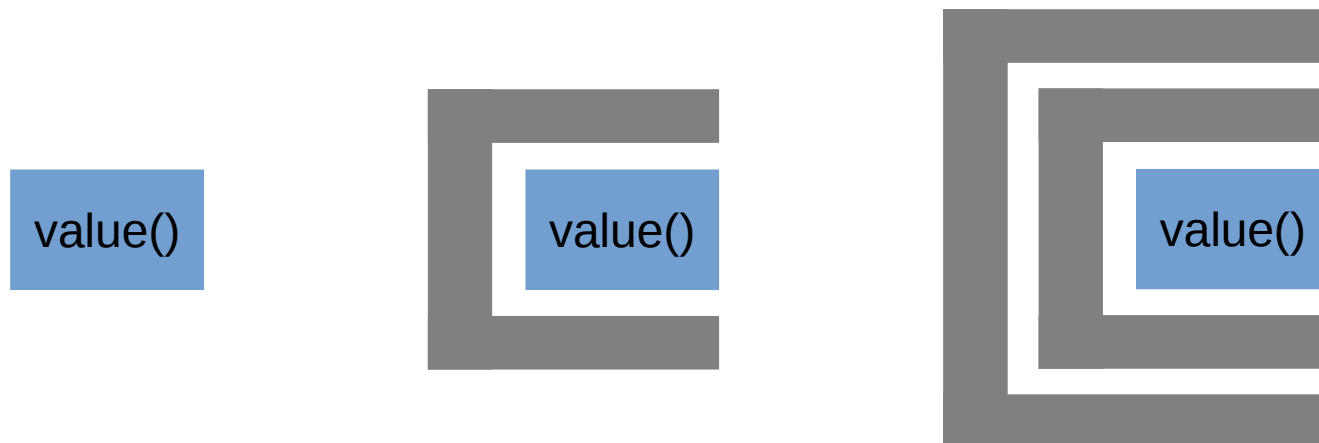
*# update the text in the widgets*

```
po.i0.value = 'i0 text'  
po.i1.value = 'new i1 text'  
po.i2.value = 'new i2 text too'
```



# IframeWrap Pattern Summary

- Use IframeWrap Pattern when you have page objects for HTML elements that work the same inside or outside of an iframe
- Uses Decorator Pattern to wrap page object attributes with calls to enter and exit iframe



# IframeWrap Pattern Gotchas

- Not all page object attributes need to be wrapped.
  - Keep a list of specific methods not to wrap.
- Wrapping Python properties can be tricky
  - Create a new class object from old class.
  - Decorate the property functions of new class.
  - Associate new class with page object.

# More Info

- Design Patterns Book
- Websites
  - <http://www.oodesign.com>
  - <http://sourcemaking.com>
- Special Thanks
  - HUBzero Team
  - Sam Midkiff, Advisor

