Pages

Introduction

Wicket uses Java classes to represent pages. Most of the time, you will create a subclass of WebPage:

MyPage.java

```
package org.wicket-wiki.example;
public class MyPage extends WebPage {}
```

And the markup:

MyPage.html

```
<html>
  <body>
     ... content ...
  </body>
</html>
```

You can reach this class under the following URL:

`http://localhost:8080/wicket/bookmarkable/org.example.MyPage`

See Newuserguide / An interactive HelloWorld for a verbose example.

Pages are objects

Because pages are Java objects, you can create new instances and pass them around like any other object. You can use the constructor, for instance:

MyPage.java

```
public class MyPage extends WebPage {
   public MyPage(String name) {
      add(new Label("name", name));
   }
}
```

Usage:

Link handler in AnotherPage.java

```
public void onClick() {
   // creating another page instance and going to that page
   setResponsePage(new MyPage("keuner"));
}
```

Because the entire page stays in the session, you do not have to put client related state in the session explicitly.

Page types

In Wicket pages can have two characteristics, they can be **bookmarkable** and / or **stateless**.

Bookmarkable pages are pages which have a constructor with no arguments or a constructor that accepts a PageParameters argument (which wraps any query string parameters for a request). In case the page has both constructors, the constructor with PageParameters will be used only if there are parameters in the URL, though you should not rely on this behavior and both constructors should do the same thing.

```
public class MyPage extends WebPage...
    public MyPage() {
        this(new PageParameters());
    }

    public MyPage(final PageParameters parameters) {
        ...
    }
```

See also Bookmarkable pages and links.

Stateless pages are pages which are bookmarkable and contain only stateless components and stateless behaviors. Unlike stateful page Wicket doesn't store stateless pages in Session (to be precise in a PageMap). See Stateless pages for more information.

Here is a simple table showing how bookmarkability and statelessness are related:

	bookmarkable	**not-bookmarkable**
stateless	may be	never
stateful	may be	may be

Page URLs

For this discussion of page URLs I found it useful to distinguish between three "types" of page URLs:

1. URL to page class (requesting it will cause page creation)
2. URL to page instance (requesting it retrieves page instance from Session where it is stored)
3. URL to component on a page (requesting it calls component event handler)

Stateful bookmarkable pages

For stateful bookmarkable pages URL to page class will look like :
`http://<...>/?wicket:bookmarkablePage=<page class>`
This type of pages can also be "mounted". Mounting means assigning a URL to bookmarkable page (not-bookmarkable pages cannot be mounted).

```
public class MyApplication extends WebApplication...
    @Override
    protected void init() {
        mountBookmarkablePage("mountedURL", MyPage.class);
        mountBookmarkablePage("login", LoginPage.class);
    }
```

For mounted pages URL to page class will look like:
`http://<...>/mountedURL`

URL to page instances for stateful bookmarkable pages will look like:
`http://<...>/?wicket:interface=<data to get page from Session>`
URL to components on the page will look like:
`http://<...>/?wicket:interface=<data to get page from Session including path to component>`

Not-bookmarkable pages

There are no URLs to page class for stateful not-bookmarkable pages (note that not-bookmarkable always implies stateful). Since non-bookmarkable pages don't have constructor which can be used by Wicket to create them, they can be only created explicitly in user code. The only way user can go to not-bookmarkable page is to be redirected from bookmarkable page with code like:

```
add(new Link("linkToNotBookmarkablePage") {
    public void onClick() {
        setResponsePage(new MyNotBookmarkablePage(... some constructor parameters ...));
    }
}
```

URL to page instances and components for not-bookmarkable pages is the same as for bookmarkable pages, i.e.:
`http://<...>/?wicket:interface=...`

Stateless bookmarkable pages

For stateless bookmarkable pages (note that stateless always implies bookmarkable) URL to page class will look the same as for bookmarkable stateful pages, i.e.:
`http://<...>/?wicket:bookmarkablePage=<page class>` or `http://<...>/mountedURL`
Since stateless pages instances are not stored in Session, there are no URLs to instances of stateless pages.
URL to components on stateless pages will look like:
`http://<...>/?wicket:bookmarkablePage=<page class>&wicket:interface=<path to component>`

This is description of URLs as they are used by default

The way URLs look for bookmarkable pages may be changed by using different "URL coding strategies" (though the logic of page creation remains the same). For example see HybridUrlCodingStrategy.

```
public class MyApplication extends WebApplication...
    @Override
    protected void init() {
        mount(new HybridUrlCodingStrategy("mypage", MyPage.class));
    }
```

Pages creation

As mentioned above, all pages are objects and can be created with java "new" keyword.

Stateful bookmarkable pages are created by Wicket (using IPageFactory) every time they are requested by URL pointing to page class. This means that you do not control how many different instances of page are created and there is no out-of-the-box solution to do that.

Not-bookmarkable page are not created by Wicket. They can only be created explicitly by user.

Stateless bookmarkable pages are created every time they are requested (though it's not something you should rely on).