FAQs

What, you want answers too?

This page addresses general FAQ's. More specific FAQ's can be found in the FAQ Category

Table of contents

Wicket architecture

What about performance and scalability?

Wicket performance is actually quite good under stress. Remember though that in many real-world application your bottleneck will be in business or database layer.

Wicket benchmark can be found here: wicket-1.3.x-benchmark.

You should also search wicket-user mailing list for performance and scalability or simply use performance-scalability-tipsas your starting point.

Versioning

Wicket stores versions of pages to support the browser's back button. Because this concept is difficult to understand, we'll present a simple use case that describes the problem and how versioning helps.

Suppose you have a paging ListView with links in the ListItems, and you've clicked through to display the third page of items. On the third page, you click the link to view the details page for that item. Now, the currently available state on the server is that you were on page 3 when you clicked the link. Then you click the browser's back button twice (i.e. back to list page 3, then back to list page 2, but all in the browser). While you're on page 2, the server state is that you're on page 3. Without versioning, clicking on a ListItem link on page 2 would actually take you to the details page for an item on page 3.

(This was taken from an IRC discussion with Martijn.)

How does Wicket know when a new browser window is opened?

Wicket uses the Javascript window.name property to detect if a new window (or tab) was opened. This property is blank by default .

How do I provide the page map for bookmarkable pages?

You do this by providing it as part of the URL. How exactly this looks depends on the 'encoding' of the request. The common cases are (myPageMap is the page map):

- a normal request, where the page map name parameter is provided as a part of the bookmarkable page request parameter:

  ```
  myapp?wicket:bookmarkablePage=myPageMap:somepackage.MyPage
  ```

- a request to a mounted URL, where the page map parameter is provided as a path encoded parameter:

  ```
  /myapp/mountedpage/param1/value1/wicket:pageMapName/myPageMap
  ```

What is the future of onAttach and onDetach methods?

Igor Vaynberg in wicket-dev:

> *We are trying to consolidate the methods. We have a bunch of internalOnAttach/internalAttach/attach/onattach methods. it's a big mess. what this refactor does is give you one method you can override - onattach() but forces the call to super.*
>
> *Doing it like it has been done doesn't work. users assume onattach() is a template method, they can override and not have to call super - but this fails if you go more then one method deep!*
>
> *if i create a custom component and do something in onattach(), then the user subclasses it and they do something in onattach() and don't bother to call super() they will break my functionality. My only choice of action is to make onattach() final in my custom component and provide yet another template for the user, onattach2() ? This just doesn't scale. Better to have a simple and clear contract - onattach and ondetach always require the call to super.*
>
> *Unfortunately the only way to do that at this point* and *keep the same method names is to do what i did.*
>
> *OT there is a JSR for software defect annotations that includes something like @MustCallSuper (forget what its called), that combined with an apt builder in an ide will make these kinds of contracts very easy to enforce at compile time. we are just not there just yet.*

How can I use wicket to develop webapps that target mobile devices?

Write clean HTML with CSS targeted toward the various mobile clients that you wish to support. More info can be found on the Mobile Devices page.

Maven

How do I skip the tests when doing a Maven build?

Add a `-Dmaven.test.skip=true` parameter to your Maven command line.

Other information about building wicket can be found on the Wicket from source page.

Is Wicket available in the central Maven 2 repository?

Yes, it is. However, we must rely on the Maven project to update their repository with the latest releases, resulting in some lag. If you need them hot and fresh, you can use the wicket-stuff's Maven 2 repository by adding the the following to your project's **pom.xml**:

```
<repositories>
    <repository>
      <id>org.wicketstuff</id>
      <name>Wicket Stuff Repo</name>
      <url>http://wicketstuff.org/maven/repository</url>
    </repository>
  </repositories>
```

Working with the Wicket API

Why are so many methods final?

Classes and methods in Wicket are generally declared as final until the need for extensibility is well understood. While this defensive approach may seem obsessive to some, the major benefit is that classes and methods which haven't been designed for extension cannot be extended until the problem(s) at hand are examined by the Wicket development team. This helps to ensure that the best approach is taken. Sometimes, straightforward extension is not the best approach. Sometimes, features of the API have been misapplied or misunderstood. It's best to catch this early.

While this does provoke more discussion and sometimes a bit of annoyance, the discussion generally improves Wicket and will help ensure greater backwards compatibility in the future. It's really a matter of how the development team manages API commitments. By making fewer commitments in the API in terms of extension points, we are more likely to be able to support those extension points that we do allow in the future. We are also more likely to catch anti-pattern use cases and provide features appropriate to the problems you are trying to solve.

Honestly, we aren't trying to annoy anyone and we do aim to respond quickly to any extension needs that arise which make sense. If you really don't agree with all this, remember that Wicket is open source. You can always check out the source code, remove the final keyword that offends you and rebuild the project.

Why no common init() method on pages?

A "two-phase init" is a pattern that deals with this and other use cases. The big downside to that pattern is that you have to transfer all your constructor arguments into fields so they can be accessed by the second phase' init method, and for a framework concerned with size of the objects this does not mesh very well. It also adds a lot more code you have to write. so as far as making this change in Wicket we won't do that. Moreover, it is simple enough to add a private init() into your basepage and forward both constructors to it.

How do I add custom error pages?

see Error Pages and Feedback Messages

How can I have images/files uploaded to and downloaded from my wicket app?

In short, use the wicket.markup.html.form.upload.FileUploadField component in your form (set as multiPart) in order to allow file uploads. Use an implementation of the wicket.markup.html.DynamicWebResource to provide downloads of the content. For a longer description with examples see Uploadin g and Downloading Files

What is the difference between setResponsePage(new MyWebPage()) and setResponsePage (MyWebPage.class)

setResponsePage(new MyWebPage()) (or **setResponsePage(new MyWebPage(myPageParameters))**) can be used if you want to have a bookmarkable url in the browser (your page must have default constructor or PageParameter constructor).
setResponsePage(MyWebPage.class) can be used if you want to pass information to pages on the serverside. This generates a session specific url (most of the time you can use hybrid url coding strategy).

The view layer, markup etc.

My markup element does not get rendered

If a tag's written like this,

```
<span wicket:id="name"/>
```

the tag's not rendered, whereas if it's written like this

```
<span wicket:id="name">name</span>
```

it is.
The rationale is that we do not automatically convert

```
<span/>
```

into

```
<span></span>
```

because that's too much "behind-the-scenes" magic for user to be happy with. As a result, as a

```
<span/>
```

has no body, `onComponentBody()` isn't called and nothing is rendered!

How can I change the location of Markup files?

Wicket reads the html template from the same location as the Page class (java class name + ".html"). Can this be changed?

Yes. See Control where HTML files are loaded from

How can I render my templates to a String?

```
public class PageRenderer extends BaseWicketTester
{
        private final Locale locale;

        public PageRenderer(Locale locale)
        {
                this.locale = locale;
        }

        public PageRenderer()
        {
                this.locale = null;
        }

        private String renderStartPage()
        {
                if (this.locale != null)
                {
                        getWicketSession().setLocale(locale);
                }

                return getServletResponse().getDocument();
        }

        public synchronized String render(Class<? extends WebPage> pageClass)
        {
                startPage(pageClass);
                return renderStartPage();
        }

        public synchronized String render(Class<? extends WebPage> pageClass, PageParameters parameters)
        {
                startPage(pageClass, parameters);
                return renderStartPage();
        }

        public synchronized String render(WebPage page)
        {
                startPage(page);
                return renderStartPage();
        }

}
```

For different approach please check Use wicket as template engine.

How to add #-anchor (opaque) to page url?

I don't know universal solution, but for mounted pages you can override particular url coding strategy to encode a parameter in special way. Say, I named parameter "#" and my strategy:

```
public CharSequence encode(IRequestTarget requestTarget) {
                if (!(requestTarget instanceof IBookmarkablePageRequestTarget))
                {
                        throw new IllegalArgumentException("this encoder can only be used with instances of " +
                                        IBookmarkablePageRequestTarget.class.getName());
                }
                IBookmarkablePageRequestTarget target = (IBookmarkablePageRequestTarget)requestTarget;

                AppendingStringBuffer url = new AppendingStringBuffer(40);
                url.append(getMountPath());
                final String className = target.getPageClass().getSimpleName();
                PageParameters pageParameters = target.getPageParameters();
                if (target.getPageMapName() != null)
                {
                        pageParameters.put(WebRequestCodingStrategy.PAGEMAP, WebRequestCodingStrategy
                                        .encodePageMapName(target.getPageMapName()));
                }
                final String fragment = pageParameters.getString("#");
                if(fragment != null)
                        pageParameters.remove("#");

                url.append("/");
                url.append(className);

                if(!pageParameters.isEmpty())
                {
                        url.append("?");
                        appendParameters(url, pageParameters);
                }
                if(fragment != null)
                        url.append("#").append(urlEncodePathComponent(fragment));
                return url;
        }
```

Now you can add parameter named "#" in PageParameters bookmarkable page link or so and it will be rendered as anchor.

More on the view: AJAX

I get errors when I use Ajax in Opera Browser (before Version Beta 9 of Opera)

If you want to use wicket's Ajax implementation with Opera Browser you have to do

```
Application.getMarkupSettings().setStripWicketTags(true);
```

to get it work.

Because of a Bug which is in all Opera Browsers before Version 9 Beta, the ajax stuff will not work with wicketTags enabled.

Keywords:
Opera Ajax Wicket Strip wicket:id

Which browsers have been tested with Wicket AJAX?

To test it use the AJAX examples.

- Google Chrome - latest two stable versions
- Firefox - latest two stable versions
- Safari 5.x and 6.x
- Opera 12.x and 15.x
- Internet Explorer 8+

Wicket.replaceOuterHtml gives weird result on Firefox

There are certain cases when you replace markup in FireFox where some of the nodes are not placed right in the DOM tree. This happens when the markup is invalid, e.g. you have somewhere a block element inside an inline element, such as <table> or <div> in <span>.

How to repaint a (List/Grid/Data/Repeating)View via Ajax?

See How to repaint a ListView via Ajax.

Wicket community

What do you look like?

You can see some of the Wicket committers in action at the Wicket gathering in 2005 in Deventer, Holland. You'll see Eelco, Johan, Juergen, Martijn and Jonathan, and emeritus committer Chris.

Miscellaneous

What is the meaning of life, the universe and everything?

42

Deployment

My application says "DEVELOPMENT MODE", how do I switch to production?

Add the following to your web.xml, inside your <servlet> definition (or <filter> definition if you're using 1.3.x):

```
<init-param>
            <param-name>configuration</param-name>
            <param-value>deployment</param-value>
</init-param>
```

You can alternatively set this as a <context-param> on the whole context.

Another option is to set the "wicket.configuration" system property to either "deployment" or "development". The value is not case-sensitive.

The system property is checked first, allowing you to add a web.xml param for deployment, and a command-line override when you want to run in development mode during development.

You may also override Application.getConfigurationType() to provide your own custom switch, in which case none of the above logic is used. See WebApplication.getConfigurationType() for the default logic used above.

How do I know in what mode (DEVELOPMENT/DEPLOYMENT) my application runs?

As of Wicket 1.3.0, you can call Application.getConfigurationType(). Prior to that, there is no flag telling you in which mode you are. If you want a flag subclass Application.configure() store the "configurationType" parameter in a variable.

```java
if (DEVELOPMENT.equalsIgnoreCase(configurationType))
{
    log.info("You are in DEVELOPMENT mode");
    getResourceSettings().setResourcePollFrequency(Duration.ONE_SECOND);
    getDebugSettings().setComponentUseCheck(true);
    getDebugSettings().setSerializeSessionAttributes(true);
    getMarkupSettings().setStripWicketTags(false);
    getExceptionSettings().setUnexpectedExceptionDisplay(
                    UnexpectedExceptionDisplay.SHOW_EXCEPTION_PAGE);
    getAjaxSettings().setAjaxDebugModeEnabled(true);
}
else if (DEPLOYMENT.equalsIgnoreCase(configurationType))
{
    getResourceSettings().setResourcePollFrequency(null);
    getDebugSettings().setComponentUseCheck(false);
    getDebugSettings().setSerializeSessionAttributes(false);
    getMarkupSettings().setStripWicketTags(true);
    getExceptionSettings().setUnexpectedExceptionDisplay(
                    UnexpectedExceptionDisplay.SHOW_INTERNAL_ERROR_PAGE);
    getAjaxSettings().setAjaxDebugModeEnabled(false);
}
```