

# Play Framework V1.x References



## **Book**

<https://www.packtpub.com/web-development/play-framework-cookbook>

1st Edition only, 2ed edition is for a different framework - Play 2



## **Book**

<https://www.packtpub.com/web-development/play-framework-essentials>



## **Video**

<https://www.packtpub.com/web-development/play-framework-web-application-development-video>

**Play Framework Cheat Sheet (reproduced on next pages)**

<https://www.playframework.com/documentation/1.2.7/cheatsheet/templates>

## Templates

[Model](#) [Command Line](#) [Tests](#) [Controllers](#) [Multi-Environment](#) [Templates](#)

### Template - Implicit objects

**errors**  
The validation errors raised in the controller

**flash**  
Flash scope

**lang**  
The negotiated language

**messages**  
The map of localised messages

**out**  
The output stream writer

**params**  
Current parameters

**play**  
Main framework class

**request**  
The current HTTP request

**session**  
The session scope

### Template - Tag grammar

**\$( client.name )**  
Evaluates and outputs a variable

**#{ client?name }**  
Displays client name only if client not null

**@( Controller.action@ )**  
Calculates URL, relative path to action

**@( Controller.action(.secure() )**  
Calculates URL, relative HTTPS path to action

**@@( Controller.action() )**  
Calculates URL, absolute path to action

**@'path/to/static/content'**  


**&( message.key )**  
Messages are maintained in conf/messages, supports i18n

**+{ this is a comment }**  
What else to say?

**%{ ou:print("HelloWorld") }**  
Groovy scripts for UI logic

**#{ myCustomTag }**  
A typical custom tag - page context not shared

### Template - Standard Tags

**#(extends 'page.html')**  
#(layout :)

Master template decorators

**#(get 'title')(used # title not set#(get)**  
**#(set title'Home Page')**  
Shared variables: between page and master templates

**#(include 'tree.html')**  
Includes fragment - page context is shared

**#(script id:'myscript', src:'script.js', charset:'utf-8')**  
**#(stylesheet id:'main', media:'print', src:'print.css')**  
Imports script & styles in the page

**#( @Application.logout() )(Disconnect#(A)**  
**#(form @Client.create(), id:'form' enctype:'multipart/form-data') ... #( /form)**  
Handy tags to create anchors and forms

**#(verbatim)#{&}#( /verbatim)**  
Disables HTML escaping

**#(i18n :)**  
Exports localized messages in javascript

**#(errors) <@>(errors) found</@> #( /ifErrors)**  
Checks for validation errors

**#(ifError 'user.name' #error'username' /) #( /ifError)**  
Checks a given error

**#(errors) <li> \$error </li> #( /errors)**  
Iterates over the current validation errors

**#(if cond)...#( /if)#(ifelse cond)...#( /ifelse)#(else)...#( /else)**  
Conditional constructs

**#(list items:0..10, as:'i')#{ /list}**  
**#(list items:'a..z', as:'i')#{ /list }#{ /last item }#{ /list}**  
List constructs

**#(loop users)#{ /list}**  
Loop constructs

**#(list item:task, as:'task')#{ /task}#{ /list}**  
**#(else)No tasks on the list#( /else)**  
Tip: else can be used along with list

**#(cache 'key', for:'15min')...#( /cache)**  
Caches content for 15 minutes

### Template - Custom Tags

**@(ast tags.Namespace("domain")**  
**public class RecaptchaTag extends Foo.Tag {**

**public static void \_recaptcha(Map args, Closure body, PrintWriter out, ExecutableTemplate template, int fromLine) { ...**  
  
**/app/view/tags/domain/mytag.tag**  
Custom tag can be called as **{@domain.mytag}**

### Template - Groovy extension

**\$( ['red', 'green', 'blue'].join(',') )**  
red,green,blue

**\$( ('red', 'green', 'blue') as String[] + 'pink').join(',') )**  
red,green,blue,pink

**\$( ('red', 'green', 'blue') as String[] + 'contains green') )**  
true

**\$( ('red', 'gr', 'blue') as String[] + 'remove gr').join(',') )**  
red,blue

**\$( ['red', 'green', 'blue'].last() )**  
blue

**\$( new Date().new Date().getTime() - 1000000).since() )**  
10 minutes ago

**\$( new Date(1275910970000).format('dd MMMM yyyy h:mm:ss')) )**  
07 June 2010 01:42:50

**\$( 1275910970000.asDate().format('dd MMMM yyyy h:mm:ss')) )**  
07 June 2010 01:42:50

**\$(725016..formatSize())**  
709KB

**\$(42..formatCurrency("EUR").raw() )**  
&euro; 42.20

**\$(42..page(10) )**  
5

**journal\$( ['om', 's+', '12'].pluralize('a', 's')) )**  
journaux

**\$( "lorum ipsum dolor" .capitalize() )**  
Lorum ipsum dolor

**\$( "lorum ipsum dolor" .capitalize() )**  
Lorum ipsum dolor

**\$( "lorum ipsum dolor" .capitalizeFirst() )**  
Lorum ipsum dolor

**\$( "lorum ipsum dolor" .capitalizeFirst() )**  
Lorum ipsum dolor

**\$( "lorum ipsum dolor" .capitalizeFirst() )**  
lor ipsum dolor

**\$( "The <blink>tag</blink> is evil" .escape().raw() )**  
The &lt;blink&gt;tag&lt;/blink&gt; is evil

**\$( "one\ntwo".n2br() )**  
one<br/>two

**\$( '<' ) \$( '<' .raw() )**  
< <

**\$( '()' ('') ^ .escape(javaScript().raw() )**  
\(\)\(\)

**\$( "'yesno'(yes,'no') )**  
no

**\$( "not empty" .yesno(yes, no) )**  
yes

**\$( "stephane epardaud" .noAccents() )**  
Stephane Epardaud

**\$( "The Play! framework's manual" .slugify() )**  
the-play-framework-s-manual

**\$( "x" .pad(4).raw() )**  
x&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&

Back

Get help

Google

Mode

Query

Access

Post

List

Find

post

Save

Mode

@Ent

Spec

@Em

Defin

Mode

@Gen

IDEN

Mode

@Tab

Defin

@id

Defin

@Ver

Defin

Mode

Conv

@Pre

Defin

Mode

@Om

Defin

@Om

Defin

Mode

@Nar

Defin

## Controller.action - Smart binding

```
Controller/link?id=32&n=patrick
public static void link(int i, String n)
public static void int(Integer i, String s)
public static void int(Long i, String n)

Controller/show?id(0) - 1&d(1) - 2&d(1) - 3&d(3) - 4
public static void show(Long id)
```

```
public static void show(List id)
public static void show(Set id)
Controller/get?date=02-18-2012
public static void get(@As("MM-dd-yyyy") Date date)
(@As(binder=MyCustomStringBinder.class))
Custom parameter binder
```

```
public static void create(String comment, File attachment)
Send File as multipart/form-data encoded POST request
?clientname=paal@client.com - p@example.com
public static void create(Client client)
Invalcan (PCO) binding
@NoBinding
Marks a non-binding field
```

## Controller.action - Validation

```
@Requires String username
@IsRequired String agree
@Max(7500) Integer wordCount
@Min(18) Long age
@MaxSize(2083) String value
@MinSize(42) String value
@email String address
```

```
@Equals("passwordConfirmation") String password
@InFuture String dueDate
@InFuture("1979-12-31") String birthDate
@Match("[A-Z](0|1)") String abbreviation
@Match("(dirac|dehlinger|cardon|beclipt)")
@Past String actualDepartureDate
@Past("1980-01-01") String birthDate
@Range(min = 17100, max = 40000) String wordCount
```

```
@URL String address
@IPv4Address String ip
@IPv6Address String ip
@Phone String phone
Related phone validation
@Valid Person person
Invalcan (PCO) validation - class Person needs validator annotations
```

## Controller - Session Management

```
WARNING: Play session is NOT the JEE session
session and flash use cookies 4KB limit, 2% cookies max
session.getId()
Returns the session ID - in most cases a must have!
session.put(String key, String value)
session.get("user_flag");
```

```
Values are limited to Strings, 4KB max
flash.put(String key, String value);
flash.get(String key);
Flash entries are discarded at end of next request
Cache.set("key_" + id, product, "30m");
Set cache value to 30 minutes
```

```
Cache.get("key_" + id, Product.class);
Get cache value, may return null
Cache.delete("key_" + id);
Non blocking cache delete
Cache.safeDelete("key_" + id);
Blocking cache delete
```

## Controller.action - Redirection

```
render(params...);
Renders template with given parameters, as text/html
renderXML(params...);
Renders parameters as application/xml
```

```
renderJson(params...);
Renders parameters as application/json
renderText(params...);
Renders parameters as text/plain
renderTemplate("Clients/showClient.html", id, client);
Bypasses default template
```

```
redirect("http://www.crisoics.com");
HTTP redirect to the given URL
From an action, calling another Controller.action()
The framework transparently generates a REDIRECT!
```

## Controller - Job

```
@OnApplicationStart
@On("0 0 12 * * ?")
Every day at 12:01pm
@On("10 30 12 ? * MON-FRI")
Every working day at 12:30pm and 10a
```

```
@Every("1h")
public class Backstrax extends Job {public void doJob() {}}
```

```
@Catch(value={RuntimeException.class})
public static void onException(RuntimeException e) {...}
Exception handling at the controller level
```

## Controller - Interceptions

```
@Before - action - @After - template - @Finally
Interceptions evaluation order
@Before static void checkAuthentications()
@After static void log()
```

```
@Finally static void audit()
You get the idea
@WebHook(Secure.class)
public class Admin extends Application
Custom interceptors at the controller scope
```

```
Query query = JPAem.createQuery("query");
Access the persistence manager
```

## Controller.action - Others

```
logger.info("Action executed ...");
logger.debug("A log message");
logger.warn("Dep");
logging configuration lives in application.conf
```

```
@Cachefor("1h") public static void index() {...}
Caches the result of the action for 1 hour
Play.configuration.getProperty("log.dir");
Access to the configuration file
```

```
Images.toBase64(File image);
Handy method!
```

## Controller - Libraries

```
WSUrl("http://s.com/posts").get().toJSON();
HTTP GET request to JSON
WSwithEncoding("iso-8859-1")
WSUrl("http://s.com/posts").get().toJSON();
HTTP GET request to JSON using iso-8859-1 encoding
WSUrl("http://s.com/").post().toXML();
HTTP POST request to XML
@B.execute("raw sql");
Eval raw SQL
XML.getDocument(String);
String to XML
XML.serialize(Document);
XML to String
```

```
XPath.evaluate(String xpath, Object node);
XPath expressions evaluator
Files.copy(File, File);
File copy
Files.copyDir(File, File);
Recursive directory copy
Files.delete(File);
Files.deleteDirectory(File);
Deletes file, directory
IO.readLine(File);
IO.readContentAsString(File);
IO.readContentFile();
IO.write(byte[], File);
Read/Write file contents
Images.crop(File orig, File to, int x1, int y1, int x2, int y2);
Images.resize(File orig, File to, int w, int h);
```

```
Crypto.encryptAES(String);
Crypto.decryptAES(String);
encrypt/decrypt using the application secret key
Crypto.passwordHash(String);
Create an MD5 password hash
Codec.UUID();
Generates unique IDs
Codec.byteToHex(String byte[]) byte[];
Write a byte array as hexadecimal String
Codec.encodeBase64(byte[] value);
Codec.decodeBase64(String base64);
Encoder/Decoder a base64 value
Coder.hexSHA1(String);
Build a hexadecimal SHA1 hash for a String
```



1 in 4

se

HW

Back to [Table of contents](#)

Get help with Google:



Play 1.2.7

## Command Line

[Model](#) [Command Line](#) [Tests](#) [Controllers](#) [Multi-Environment](#) [Templates](#)

### Command line - play command

#### classpath

Display the computed classpath

#### id

Define the framework ID, used for multi-environment configuration

#### secret

Generate a new secret key, used for encryption

#### install

Install a module

#### list-modules

List modules available in the central module repository

#### modules

Display the computed modules list

#### new

Create a new application

#### new-module

Create a module

#### build-module

Build and package a module

#### eclipsify

Create all Eclipse configuration files

#### netbeansify

Create all NetBeans configuration files

#### idealize

Create all IntelliJ Idea configuration files

#### javadoc

Generate your application Javadoc

#### auto-test

Automatically run all application tests

#### clean

Delete temporary files (including the bytecode cache)

#### test

Run the application in test mode in the current shell

#### precompile

Precompile all Java sources and templates to speed up application start-up

#### war

Export the application as a standalone WAR archive

#### run

Run the application in the current shell

#### start

Start the application in the background

#### stop

Stop the running application

#### restart

Restart the running application

#### status

Display the running application's status

#### out

Follow logs/system.out file

#### pid

Show the PID of the running application

#### check

Check for a Play framework release newer than the current one

#### help

Display help on a specific command

Back to [Table of contents](#)

Get help with Google:



Play 1.2.7

## Multi Environment

[Model](#) [Command Line](#) [Tests](#) [Controllers](#) [Multi Environment](#) [Templates](#)

### Multi-environment

#### # Test configuration

%test.db=mem

%test.jpa.ddl=create-drop

#### # Dev configuration

%dev.http.port=8080

%dev.application.log=DEBUG

%dev.application.mode=dev

#### # Production configuration

%prod.http.port=80

%prod.application.log=INFO

%prod.application.mode=prod

play run --%prod

play run --%dev

play test

Will pick the appropriate configuration



Back to [Table of contents](#)

Get help with Google:

## Tests



Play 1.2.7

[Model](#) [Command Line](#) [Tests](#) [Controllers](#) [Multi-Environment](#) [Templates](#)

### Test - Unit Tests

```
@Test public void getRental() { ... }
@Test (expected = NumberFormatException.class )
@Ignore
Ignore any errors
@Test (timeout=400)
Test will fail after 400 milliseconds
```

```
@Before public void prepareRecords();
Run before each unit test

@After public void cleanupJunk()
Run after each unit test

@BeforeClass void whenTestClassInstantiated();
Run once, when the test class gets instantiated
```

```
@AfterClass void whenTestClassCompleted()
Run once when the test class is dismissed

Assert.assert
There are tons of assertions, look it up
```

### Test - Functional Tests

```
public class ApplicationTest extends FunctionalTest
Functional test are unit tests with an HTTP orientation
```

```
newRequest()
newResponse()
GET(request, url)
PUT(request, url)
POST(request, url)
DELETE(request, url)
```

```
assertStatus(404, response)
assertHeaderEquals(headerName, value, response)
assertContentEquals(content, response)
assertContentType(type, response)
clearCookies()
```

### Test - Selenium Tests

```
@(selenium)
clearSession()
open('/admin')
```

```
assertTextPresent('login')
type('login', 'admin')
type('password', 'secret')
clickAndWait('signin')
```

```
// Verify that the user is correctly logged in
assertText('success', 'Welcome admin!')
@(selenium)
```

### Test - Data loader

```
@Before public void setUp() { Fixtures.deleteAll();
Fixtures.load("data.yml"); }
```

```
Fixtures is used to initialise the datastore before running a unit test
@(fixture delete:'all', load:'data.yml' /)
@(selenium) ... @(selenium)
```

Same idea using a Selenium test