Play! 1.0.2.1 cheat sheet

Java extensions - convenience methods on view template expressions

array.add(value)
Adds *value* to the end of the array.

string.addSlashes()
Backslash-escapes single and double quotes.

map.asAttr()
Formats the map as HTML attributes.

map.asAttr(condition)
Formats HTML attributes if the condition is true.

timestamp.asdate(format)
Formats a *long* timestamp as a date.

timestamp.asdate(format, language)
Formats a *long* as a date in the given language.

string.asXml()
Parses the given XML string.

string.camelCase()
Formats the string in camel case, e.g. *lnCamelCase.*

object.capAll()
Capitalises every word in the object's *toString()*.

object.capFirst()
Capitalises the first word in the object's *toString()*.

string.capitalizeWords()
Capitalises every word in the string.

array.contains(string)
Returns *true* if the *String* array contains the given string.

string.cut(substring)
Removes occurrences of the substring from the string.

number.divisibleBy(divisor)
Returns *true* if the number is divisible by the divisor.

object.escape()
HTML-escapes the object's *toString()*.

string.escapeHtml()
Escapes reserved HTML characters.

string.escapeJavaScript()
Escapes reserved JavaScript characters.

string.escapeXml()
Escapes reserved XML characters.

date.format(format)
Formats the date using the pattern.

date.format(format, language)
Formats the date using the pattern and language.

number.format(format)
Formats the number using the pattern.

number.formatCurrency(currencyCode)
Formats the number as the given currency.

long.formatSize()
Formats a number of bytes as a file size, with units.

collection.join(separator)
Formats the collection with a separator between items.

collection.last()
Returns the last item in the collection.

string.nl2br()
Replaces newline characters with HTML *
* tags.

string.noAccents()
Removes accents from the letters in the string.

string.pad(length)
Pads the string with * * up to the given length.

number.page(pageSize)
Returns a page number for the given item number.

collection.pluralize(), number.pluralize()
Returns an 's' when the collection size/number is not 1.

collection.pluralize(plural), number.pluralize(plural)
Returns the plural for the collection size or number.

collection.pluralize(forms),number.pluralize(forms)
Returns the plural form for the collection or number.

object.raw()
Returns the object without template escaping.

object.raw(condition)
Returns the unescaped object if the condition is true.

array.remove(string)
Returns the array, with the string removed.

date.since()
Formats the date as how long ago before now.

date.since(condition)
Like *since()*, but only up to 1 month if condition is true.

string.slugify()
Formats the string as a 'slug' for use in URLs.

string.urlEncode()
URL-encodes the string, for use in query strings.

object.yesNo('yes', 'no')
Returns 'yes' if the object evaluates to *true,* or 'no'.

Command line - play *command*

auto-test
Automatically run all application tests

build-module
Build and package a module

classpath
Display the computed classpath

clean
Delete temporary files (including the bytecode cache)

eclipsify
Create all Eclipse configuration files

help
Display help on a specific command

id
Define the framework ID

idealize
Create all IntelliJ Idea configuration files

install
Install a module

javadoc
Generate your application Javadoc

list-modules
List modules available from the module repository

modules
Display the computed modules list

netbeansify
Create all NetBeans configuration files

new
Create a new application

new-module
Create a new module

out
Follow logs/system.out file

pid
Show a running application's process ID

precompile
Precompile all Java sources and templates

run
Run the application in the current shell

restart
Restart the running application

secret
Generate a new secret key

status
Display the running application's status

start
Start the application in the background

stop
Stop the running application

test
Run the application in test mode in the current shell

war
Export the application as a standalone WAR archive

Template tags - #{*tag parameters*} … #{/*tag*} - #{*emptyTag* /}

'…' indicates a tag body, e.g. *#{if condition} … #{/if}*

a @Application.logout() …
HTML link to a controller action.

doLayout
Insert sub-template contents.

else …
After *if*, when false. After *list,* when empty.

elseif tasks.size() == 0 …
Evaluate the tag body if the condition is true.

extends 'main.xhtml'
Extend from another template.

field 'user.name'
Define *field,* with *id, name, value, error* properties.

form @Client.create(), id:'creationForm' …
HTML form; method and URL from routes.

get 'title'
Retrieve value defined with *set* tag.

i18n
Export localised messages in JavaScript.

if user.countryCode == 'en' …
Evaluate the tag body if the condition is true.

ifnot tasks …
Evaluate the tag body if the condition is false.

include 'tree.html'
Include another template.

jsAction @Users.show(':id')
Return a JavaScript function for a server action.

list users, as:'user'
Iterate over a collection.

set title:'Admin'
Define a value for use with the *get* tag.

Also refer to the complete Play framework documentation online at http://www.playframework.org/documentation