Catalyst Quick Reference Card
Revision 0.06 for Catalyst version 5.30
Andrew Ford

Catalyst is an elegant web application framework written in Perl (see catalyst.perl.org) and the Catalyst module on CPAN. The information in this refcard is abbreviated and simplified – refer to the documentation for full details.

Creating a Catalyst Application
Catalyst includes a helper script to bootstrap a new application:

```perl
$catalyst = \{
$MyApp = \{
$catalyst.pl

This creates a directory named `My-App` with skeleton module, build, and test files and a number of helper scripts, including one to launch a test server:

```bash
$ cd My-App
$ script/my_app_server.pl
```

You can connect to your server at `http://host.domain:3000/`

Application Structure
The generated application module is structured like this (with comments and documentation omitted for brevity):  
```perl
package My::App;
use Catalyst qw/-Debug/;

My::App->config(name => 'My::App');
My::App->setup;

sub default: Private {
    my ($self, $c) = 0;
    $c->res->output('Congratulations, '
        'My::App is on Catalyst!');
}
```

Catalyst loads any component modules it finds within the subdirectories: `M` (or `Model`), `V` (or `View`), or `C` (or `Controller`).

catalyst.pl creates another helper script for generating skeleton components modules:

```bash
$ script/my_app_create.pl {M|V|C} Class \ {helper arg ...}
```

Actions
The behaviour of a Catalyst application is defined by its actions, subroutines marked with attributes defined in the application or component modules:

```perl
sub action1: Global { ... }
    top-level action mapped to application base URL

sub action2: Local { ... }
    'namespace-prefixed' – sub name prefixed by component URL
```

```perl
sub action3: Path('path') { ... }
    literal action matches specified URL

sub action4: Regex('regex') { ... }
    globally matches URLs that match regex
```

There are four built-in private actions:

```perl
sub begin: Private { ... }
    special action run at start of request before matching actions

sub auto: Private { ... }
    called if no other action matches

sub default: Private { ... }
    called at start of request after matching actions

sub end: Private { ... }
    called at end of request after all matching actions
```

Context methods
Each action subroutine is invoked with two parameters – the component object reference or class name and a context object (conventionally designated `$_`).

```perl
$obj = $c->config
$obj = $c->forward(Saction, @args)
$obj = $c-$log($newobj)
$obj = $c-$plugin($name, $class, @args)
$req = $c-$request # or $c-$req
$resp = $c-$response # or $c-$res
$href = $c-$stash
```

Methods for plugin writers

```perl
$obj = $c-$dispatcher
$obj = $c-$engine
$aref = $c-$get_action($name, $namespace, $inherit)
```

Request methods

```perl
$req = $c-$request # or $c-$req
$str = $req-$action
$str = $req-$address
$str = $req-$arguments # or $req-$args
$str = $req-$base
$str = $req-$body
$str = $req-$content_encoding
$str = $req-$content_length
$str = $req-$content_type
$str = $req-$cookie($name)
@ary = $req-$cookies
$str = $req-$header
@ary = $req-$headers
$str = $req-$hostname
$str = $req-$input
$str = $req-$match
$str = $req-$method
$val = $req-$param($name)
$href = $req-$parameters # or $req-$params
$str = $req-$path
```

Utility Functions (Catalyst::Utils)

```perl
$aref = attrs($Scoderef)
stacles = class2appclass($class)
```

Plugins

```perl
Plugin methods
log methods
Utility Functions (Catalyst::Utils)
```