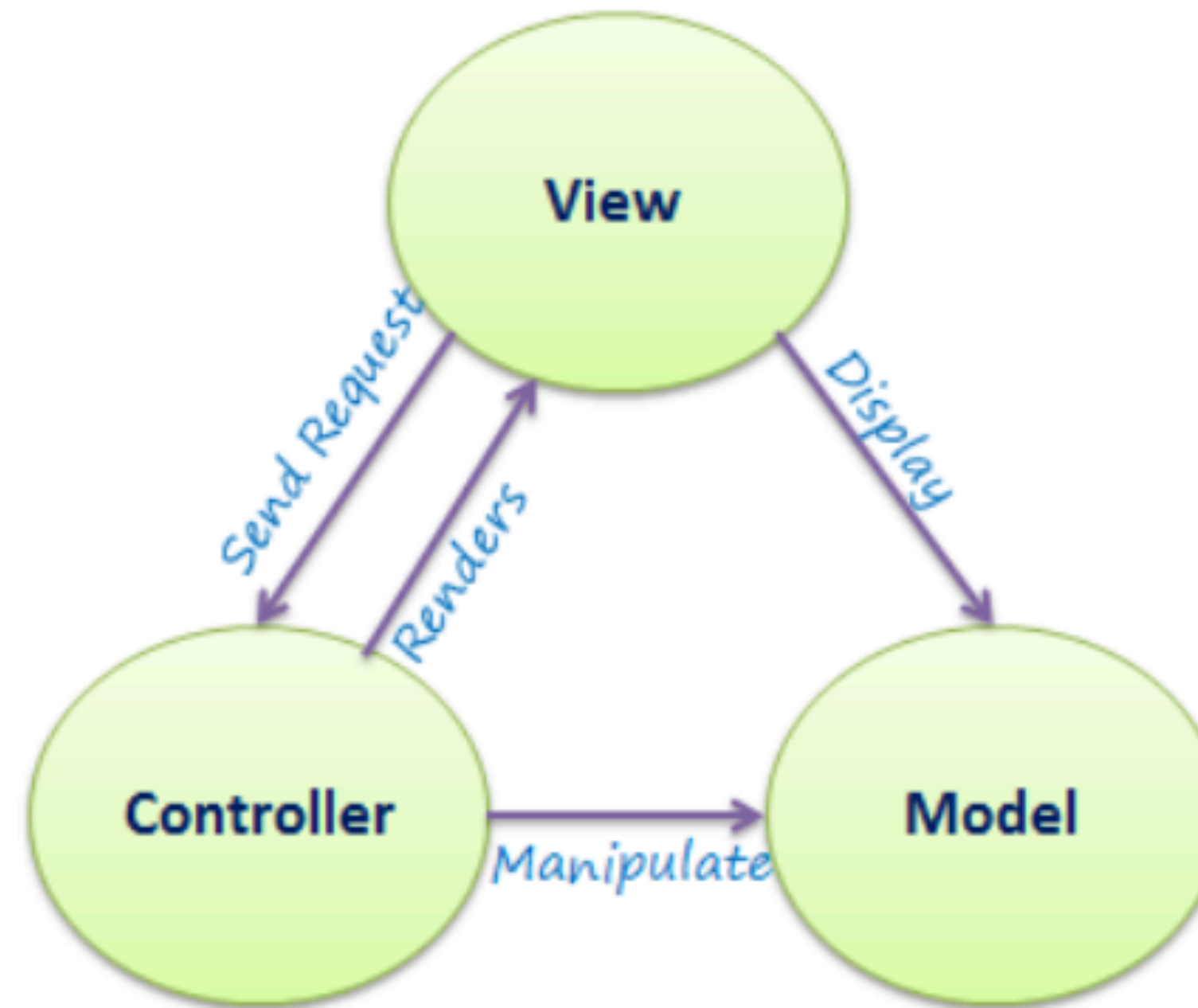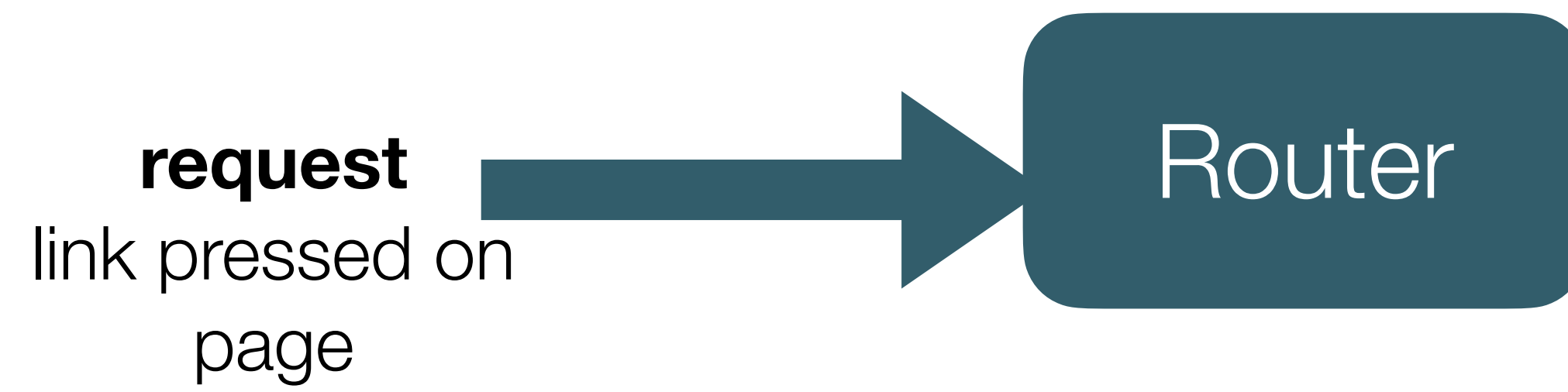# Model View Controller

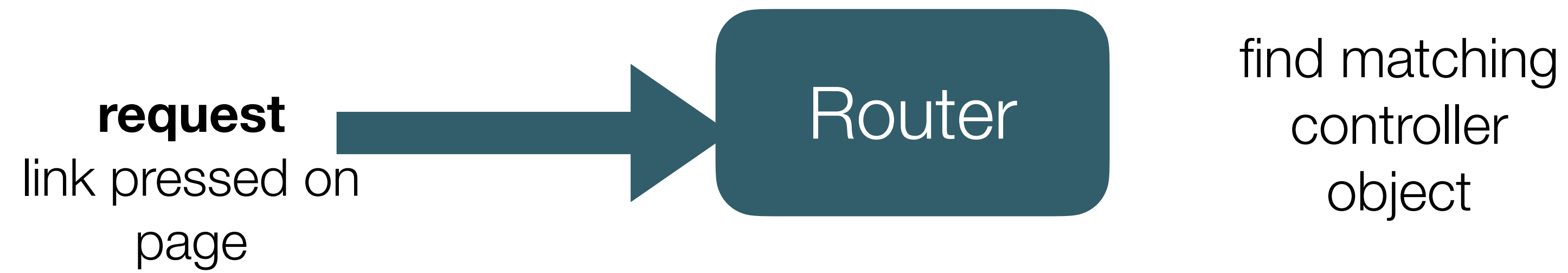request · router · controller · view · response

Request/Response Lifecycle

1. **Request** - link pressed on page

2. **Router** - find matching controller object

3. **Controller** - method invoked to handle request

4. **View** - data sent from controller to view to construct response

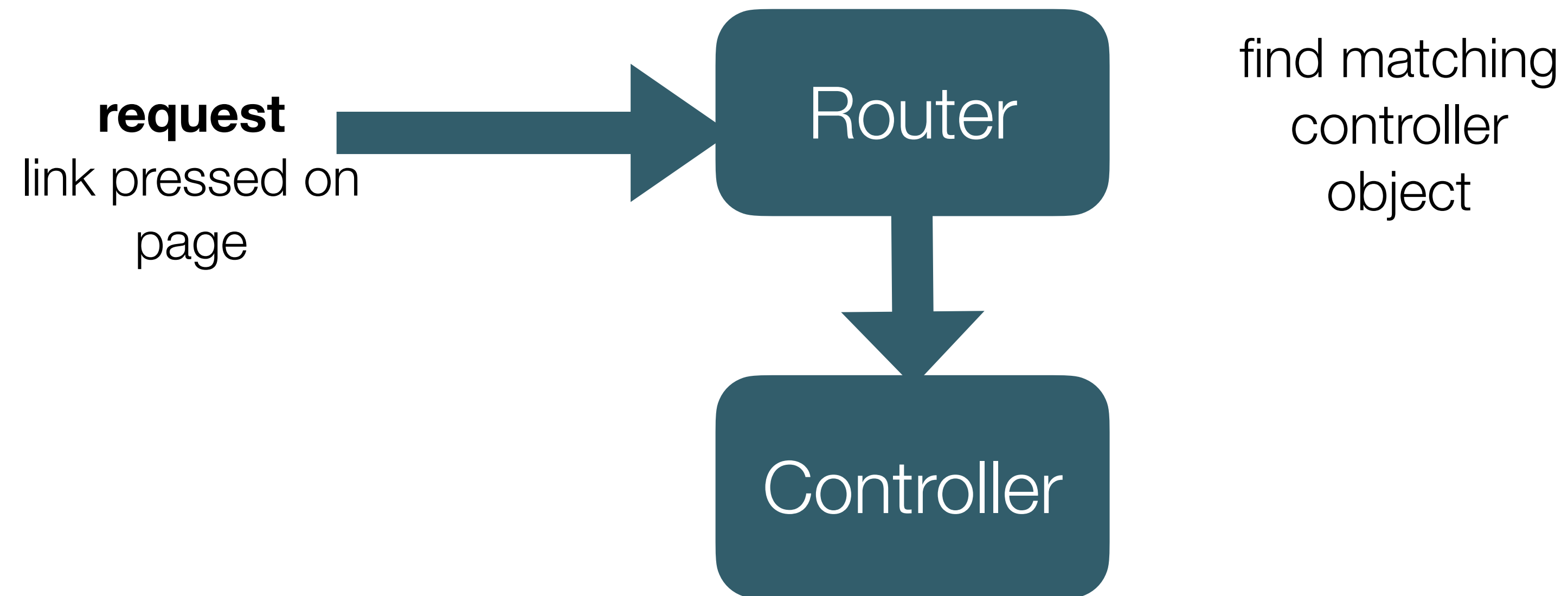5. **Response** - complete page rendered into browser

Router/Controller/View

request
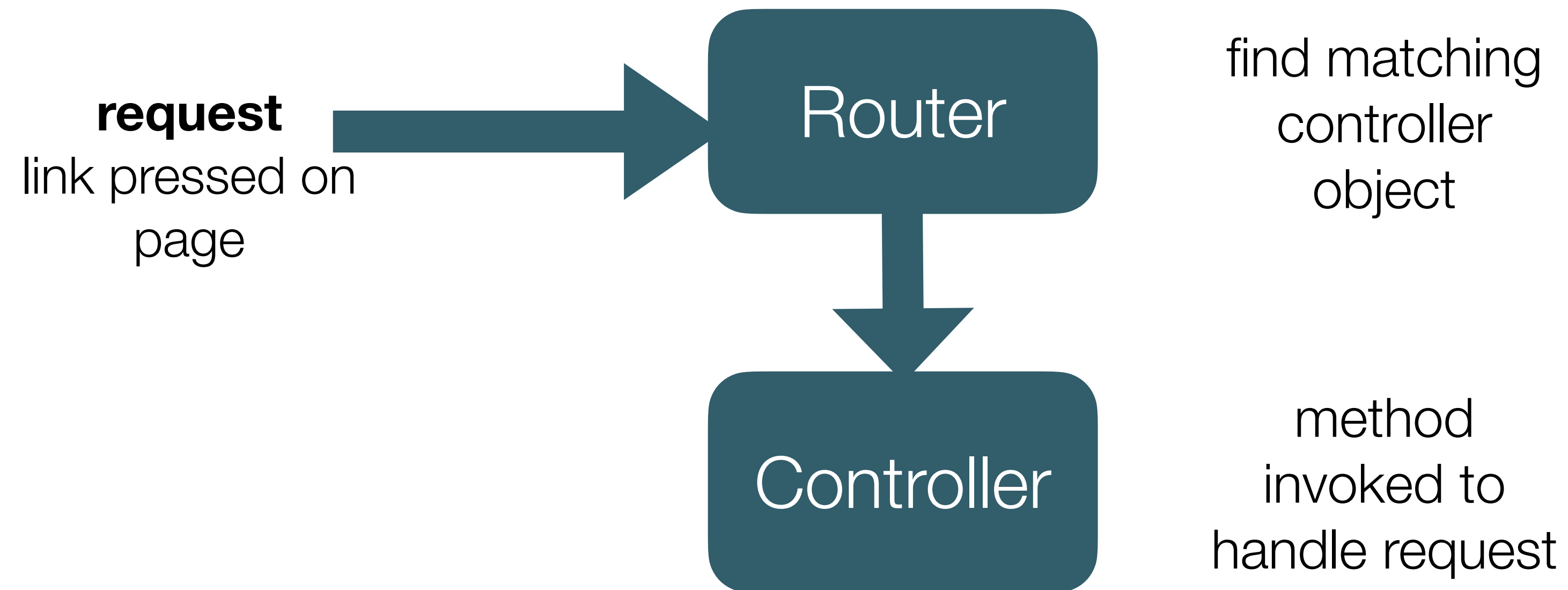link pressed on
page

Router

Router/Controller/View

request
link pressed on
page

Router

find matching
controller
object

Router/Controller/View

request
link pressed on
page

Router

find matching
controller
object

Controller

Router/Controller/View

request
link pressed on page

Router

find matching controller object

Controller

method invoked to handle request

Router/Controller/View

request
link pressed on
page

Router

find matching
controller
object

Controller

method
invoked to
handle request

View

Router/Controller/View

request
link pressed on
page

Router
find matching controller object

Controller
method invoked to handle request

View
data sent from controller to view to construct response

Router/Controller/View

request
link pressed on
page

Router

find matching
controller
object

Controller

method
invoked to
handle request

View

data sent from
controller to view
to construct
response

response
complete page
rendered into
browser
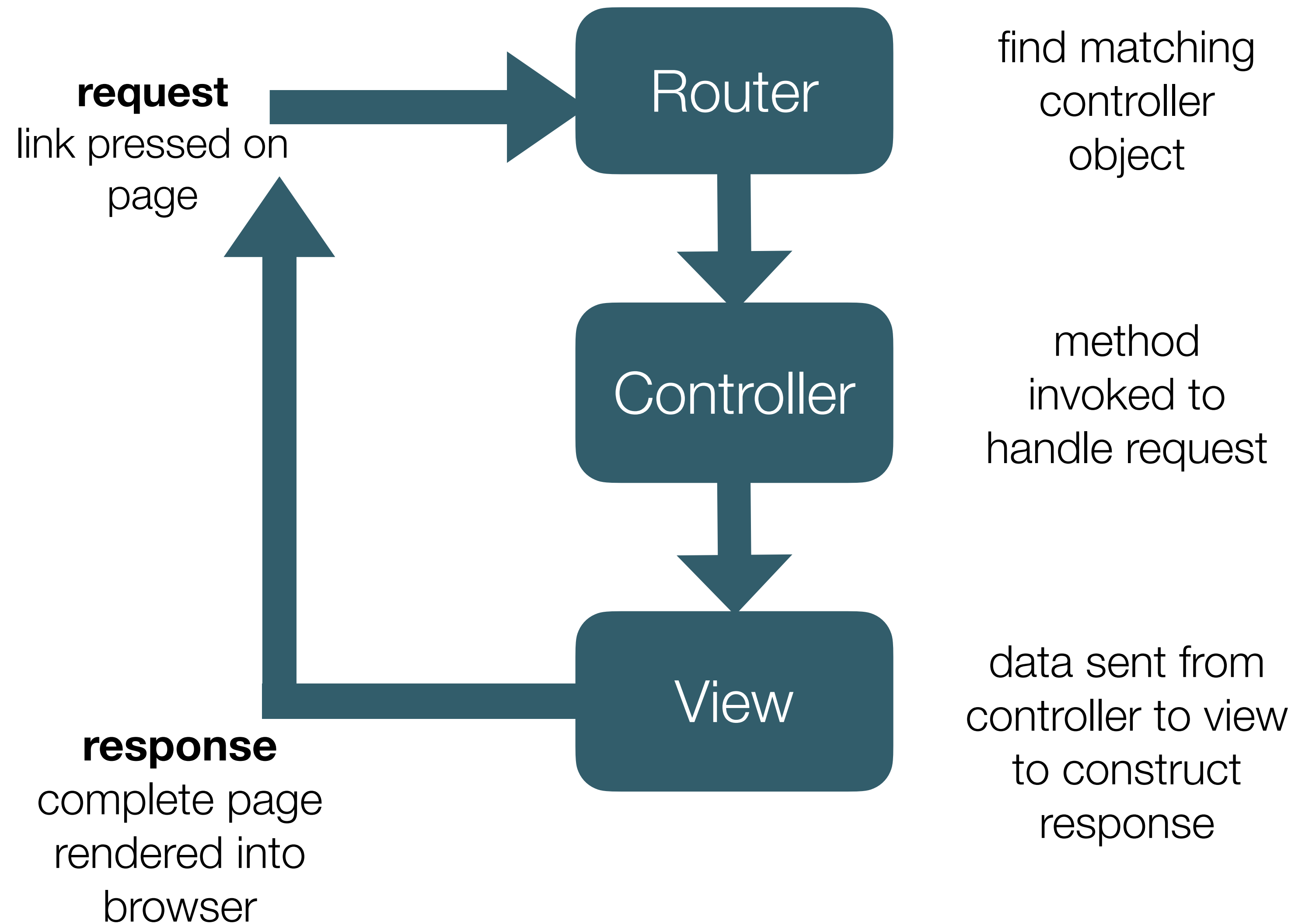
Request - link pressed on page

```html
...
    <a id="dashboard" class="item" href="/dashboard"> Dashboard  </a>
    <a id="about" class="item" href="/about"> About </a>
...
```
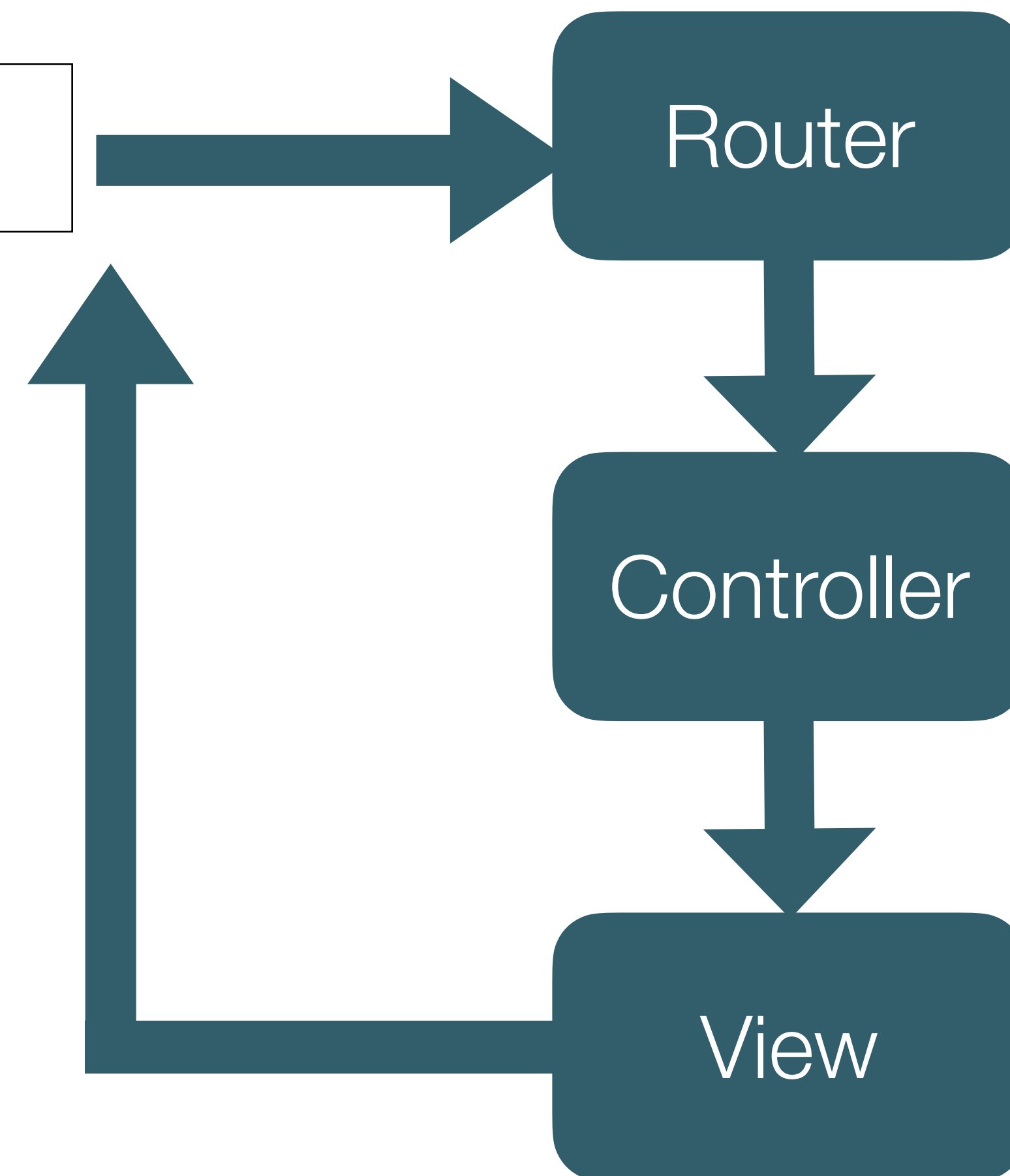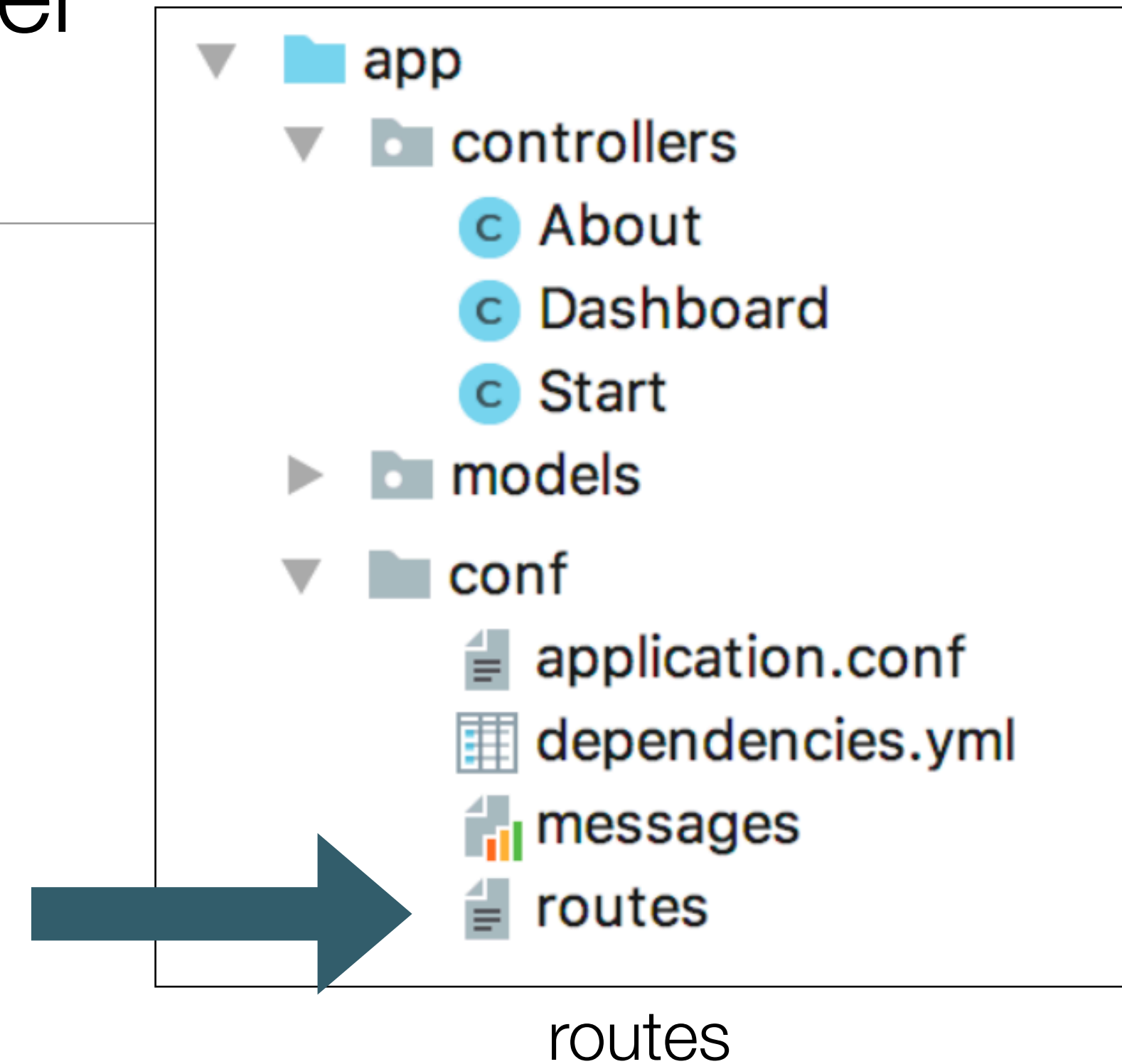
- Requests defined in links in views:

  - href in \<a\> tags

  - href in Menus

  - href in Buttons

  - action links in forms

Router - find matching controller object

routes

Mach the link patterns with the controler.method

```
...
GET     /                                              Start.index
GET     /dashboard                                     Dashboard.index
GET     /about                                         About.index
...
```
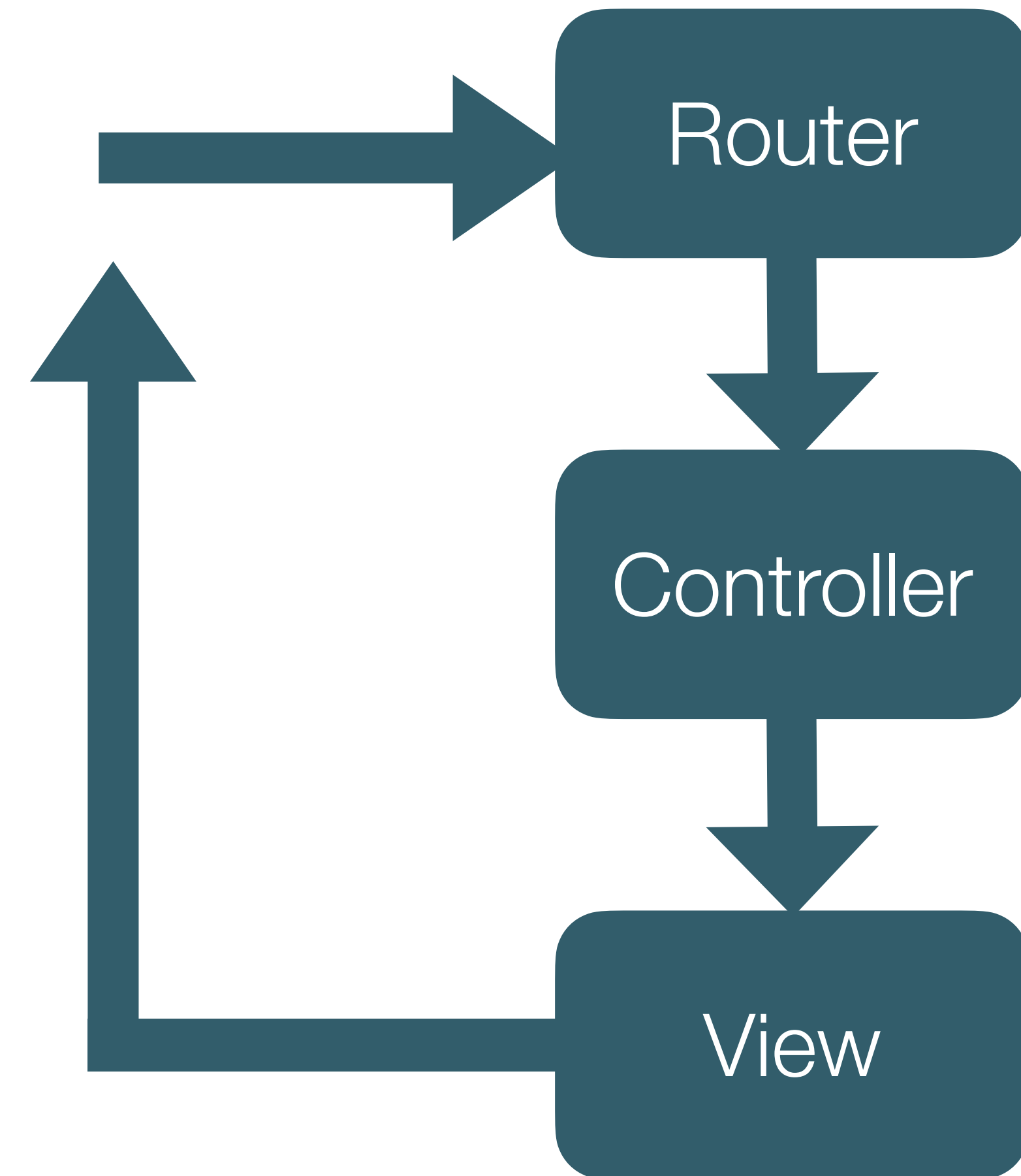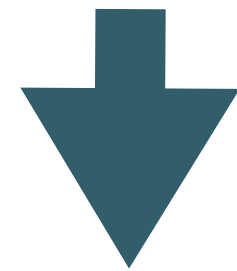
Router Behaviour

If user selects these links…

…then call the corresponding controller methods

`'/'` ⟶ `Start`.`index`

`/dashboard'` ⟶ `Dashboard`.`index`

`/about'` ⟶ `About`.`index`

Router → Controller → View → Router

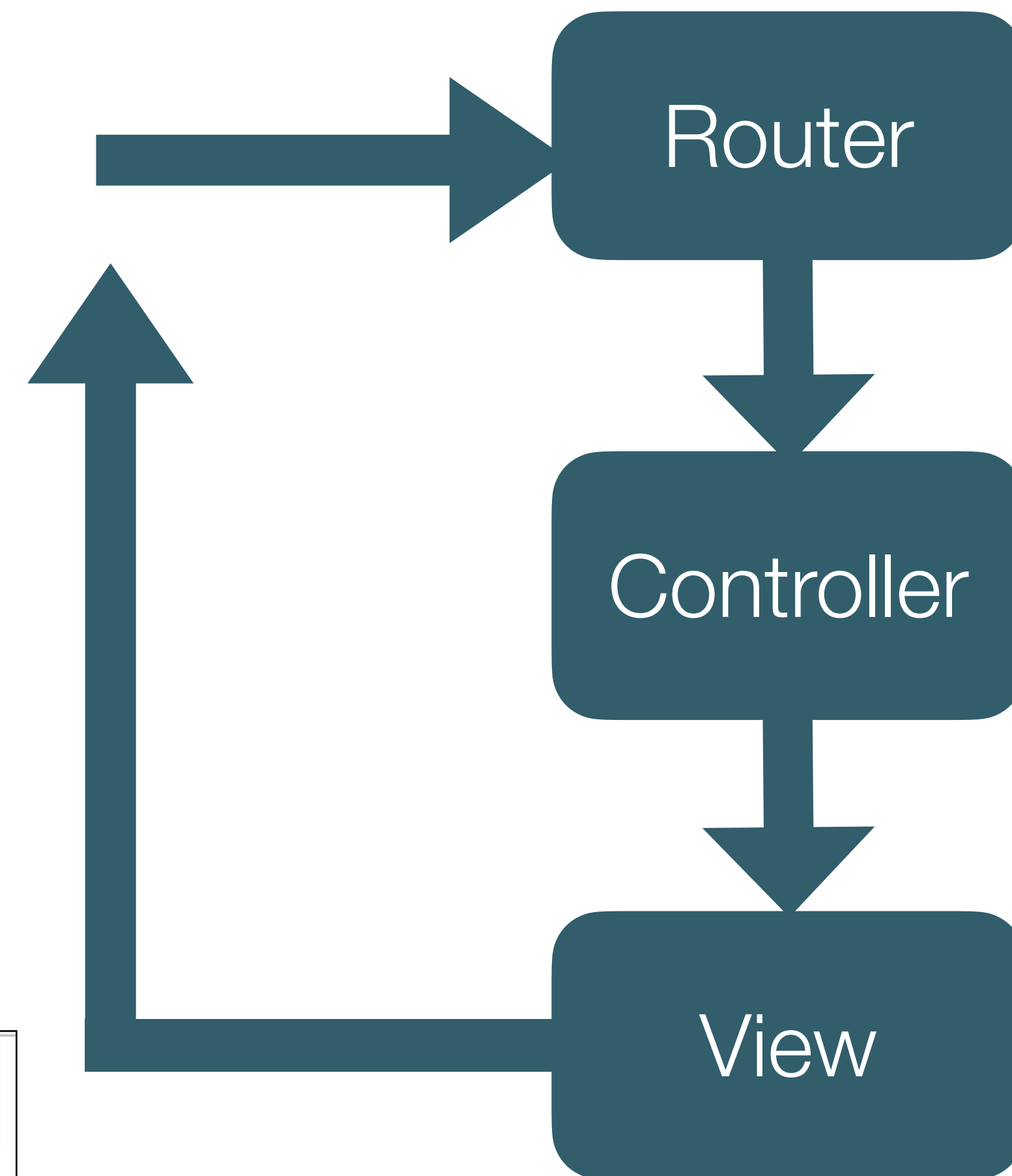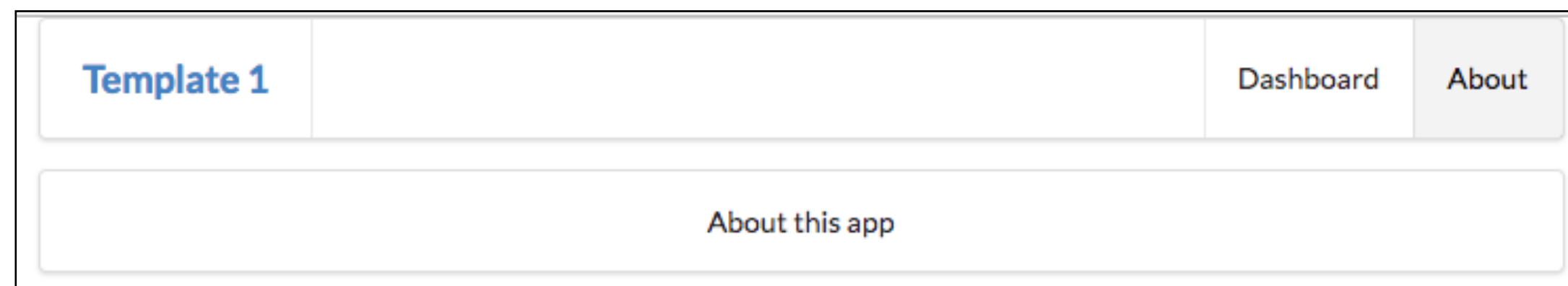# **Controller** method invoked to handle request

About.java

```java
package controllers;

import play.*;
import play.mvc.*;
import java.util.*;
import models.*;

public class About extends Controller
{
  public static void index() {
    Logger.info("Rendering about");
    render ("about.html");
  }
}
```
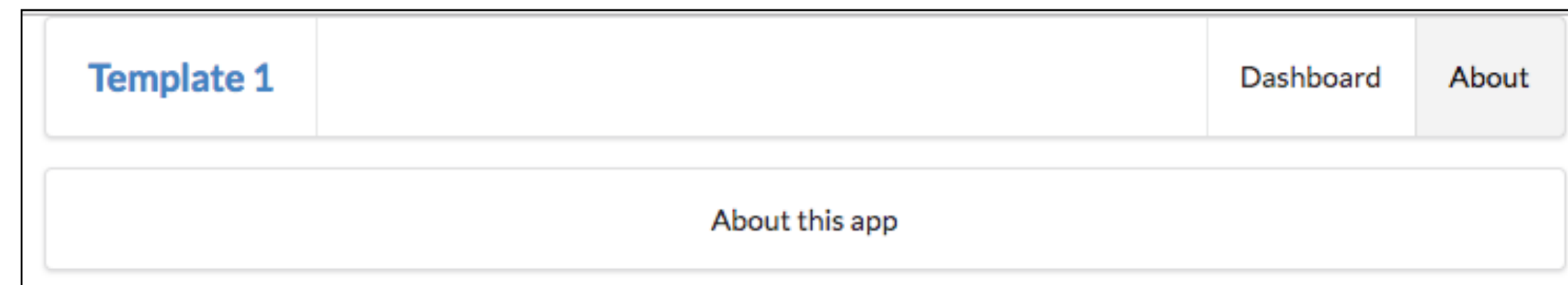
The About controller

Template 1			Dashboard	About
		About this app		

Router

Controller

View

The 'About' controller index *action*

```java
public static void index() {
    Logger.info("Rendering about");
    render ("about.html");
}
```


Log a message
to the console

Render the 'about' view

about.js

Back-end + Front-End

about.html

```
public static void index() {
    Logger.info("Rendering about");
    render ("about.html");
}
```

```
#{extends 'main.html' /}
#{set title:'About' /}

#{menu id:"about"/}

<section class="ui center aligned middle aligned segment">
    <p>
        About this app
    </p>
</section>
```

menu.html

main.html

```
<nav class="ui menu">
  <header class="ui header item"> <a href="/"> Template 1 </a></header>
  <div class="right menu">
    <a id="dashboard" class="item" href="/dashboard"> Dashboard  </a>
    <a id="about" class="item" href="/about"> About </a>
  </div>
</nav>

<script>
  $("#${_id}").addClass("active item");
</script>
```
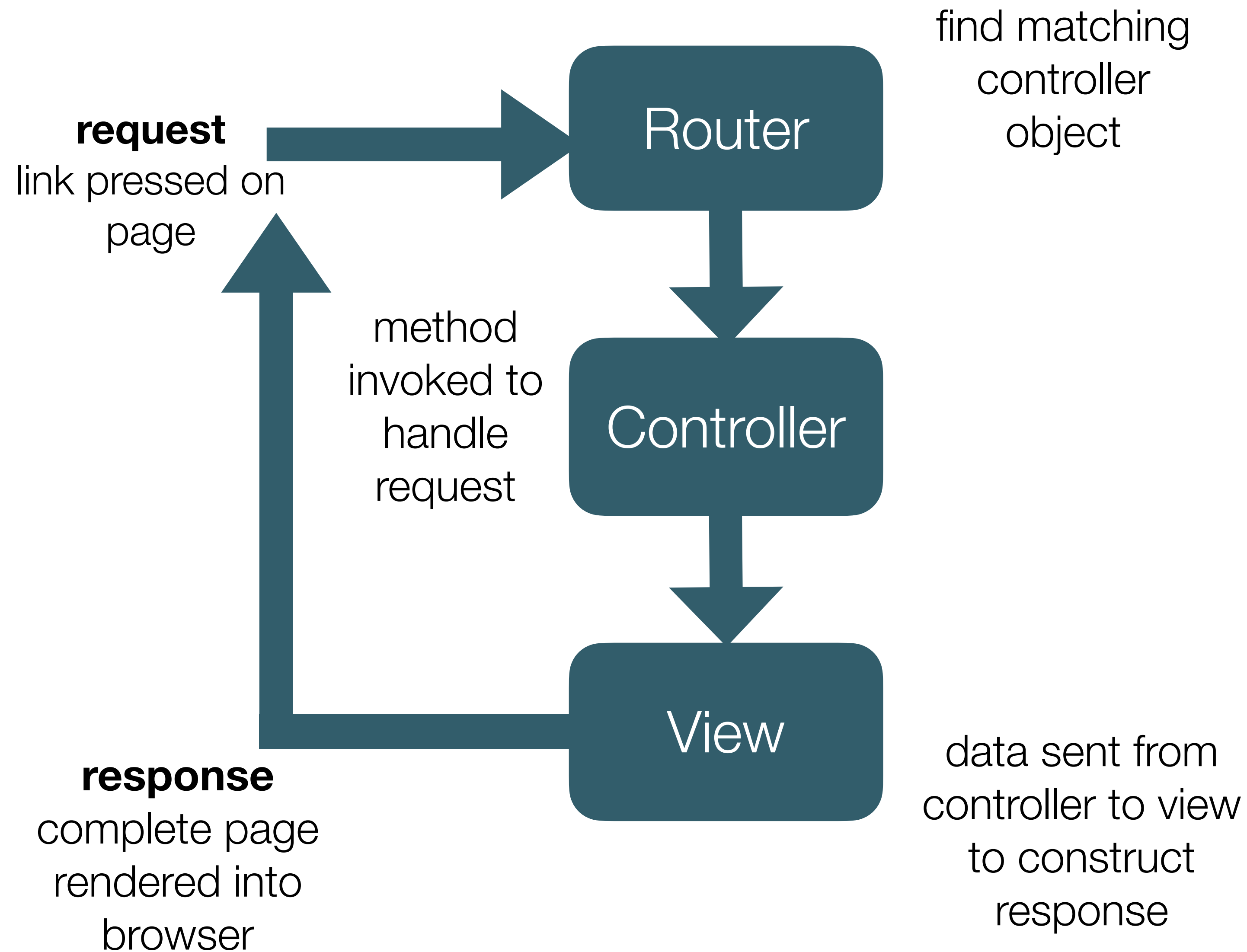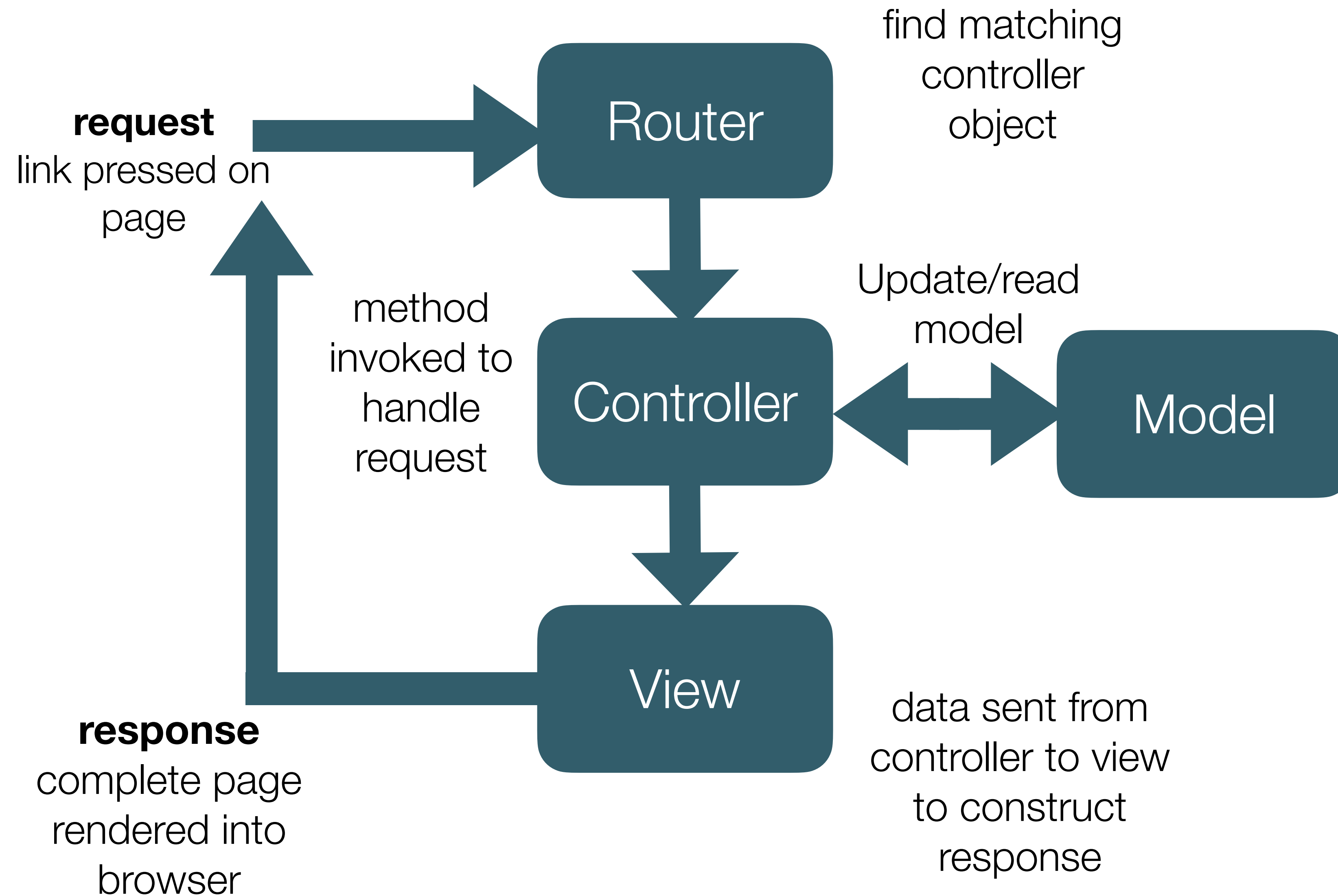
```
<!DOCTYPE html>
<html>
  <head>
    <title>#{get 'title' /}</title>
    <script type="text/javascript" src="https://cdnjs.cloudfl
    <script type="text/javascript" src="https://cdnjs.cloudfl
    <link type="text/css"  href="https://cdnjs.cloudflare.com
  </head>
  <body>
    <section class="ui container">
      #{doLayout /}
    </section>
  </body>
```

Router/Controller/View

request
link pressed on
page

Router

find matching
controller
object

method
invoked to
handle
request

Controller

View

response
complete page
rendered into
browser

data sent from
controller to view
to construct
response

Router/Controller/Model/View

request
link pressed on page

Router

find matching controller object

method invoked to handle request

Controller

Update/read model

Model

response
complete page rendered into browser

View

data sent from controller to view to construct response

Model View Controller

request · router · controller ·
view · response