CIS 197

Front End Web Development

Today

1. Git branches for HW, Quiz Review
2. Backbone library components
3. Models, Views, templates
4. Model persistence & routing

Code in lectures repo: *github.com/cis197s14/lectures*

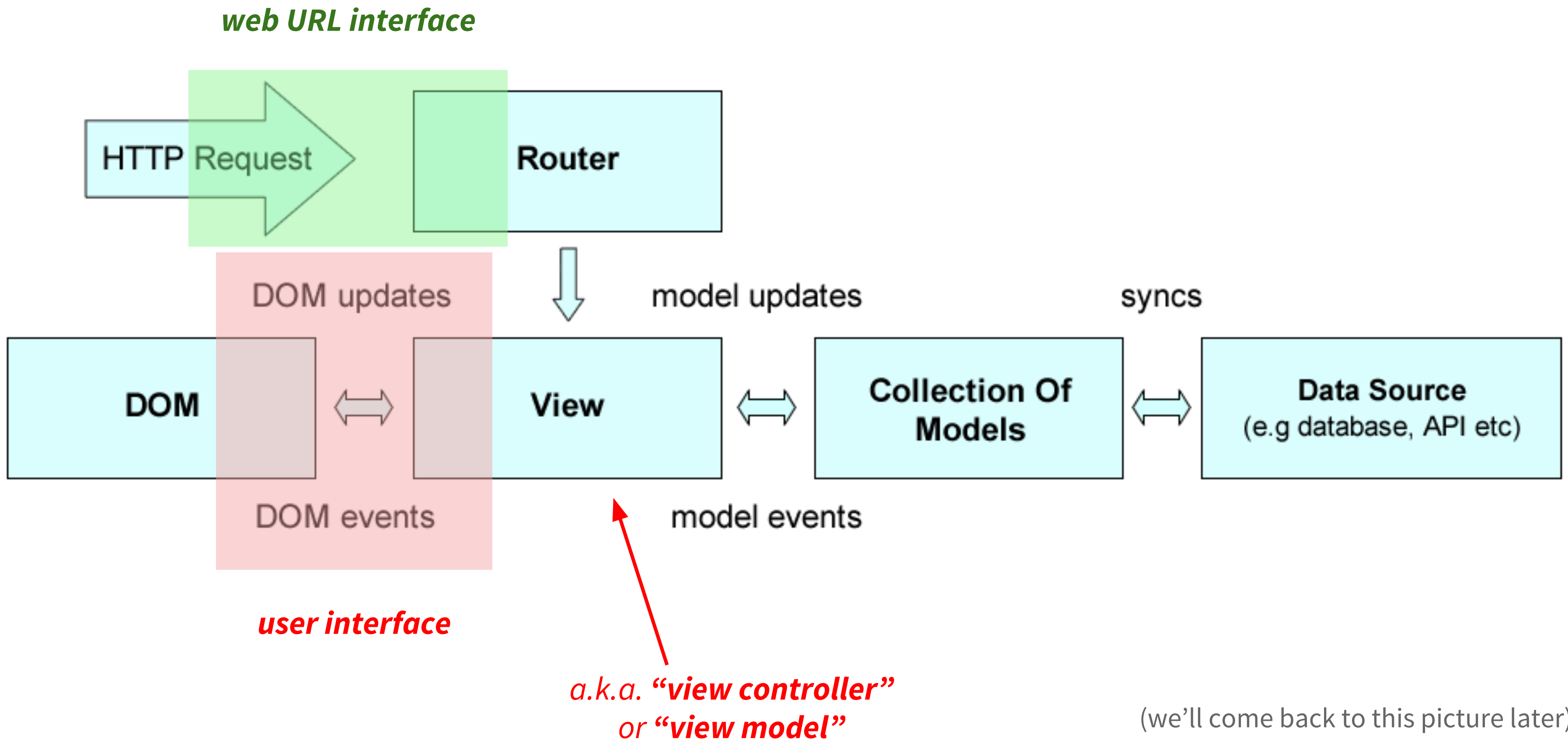
Quiz Review

1. **Purpose of an IIFE**
   - avoid polluting global namespace
2. **Function Invocation** vs. **Method Invocation**
   - `this` context is global vs. bound to object
3. **Prototype lookup chain**
   - draw it

Why Backbone or MV*?

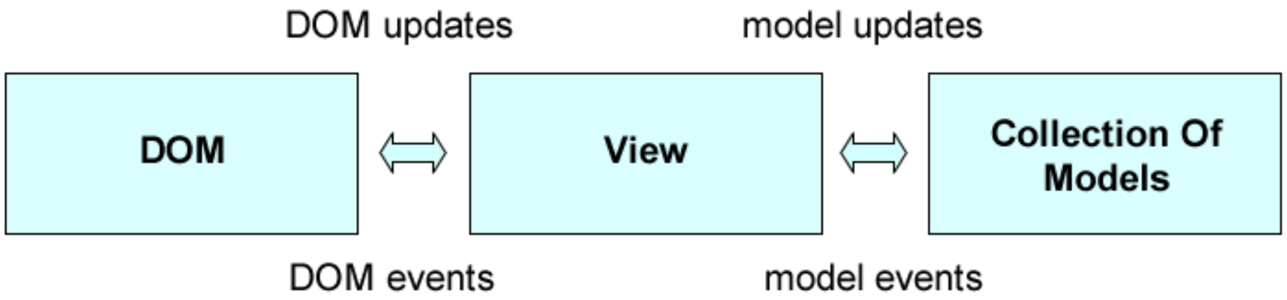
In other UI programming contexts, you need it ***to do anything consequential***. (eg. iOS)

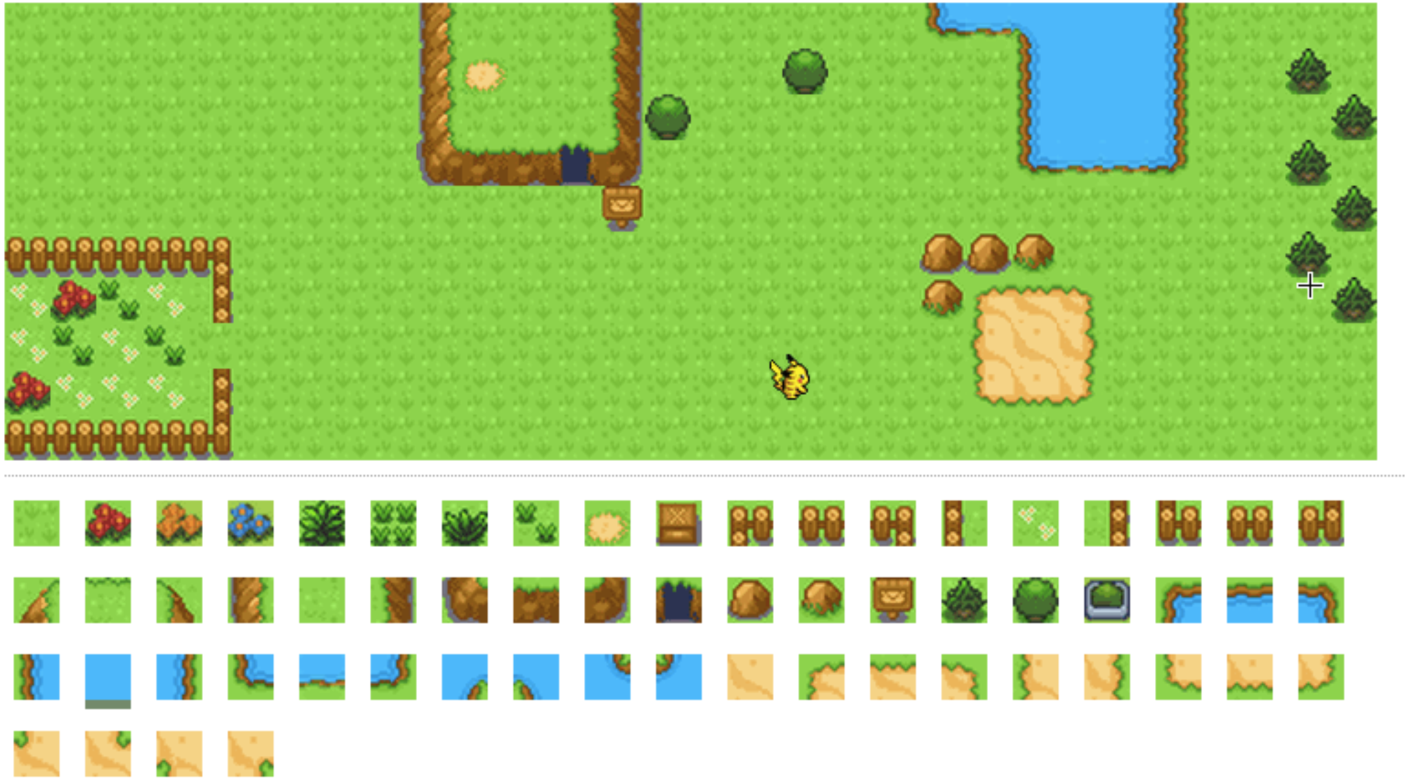
In JS, you can do consequential things without it, but ***you'll end up reinventing the wheel***.



(we'll come back to this picture later)

for now, just this part





How do we make this a Backbone app?

Models

maintain application state.

`Builder`

- selected swatch
- state of each map tile
    - swatch name?
    - is it terrain? obstruction?

`Player`

- location
- orientation

Views

display the UI & manage interactions.

`BuilderView`

- draw map tiles in DOM
- react to mouse events on tiles
    - preview swatches
    - paint swatches

`PaletteView`

- draw palette swatches
- react to mouse events on swatches

`PlayerView`

- draw & position player in map
- react to keyboard events on document

Basics

Creating Models

- `get` / `set` attributes
- `defaults`, `validate`
- listening for changes with Events

Creating Views

- view lifecycle
- attaching to an `el`
- `render`
- listening for model events

Client-side Templating

- what's in a template
- "view context"

What happens when a view is created?

1. `element` created (if not provided)
2. `events` delegated
3. `initialize` invoked

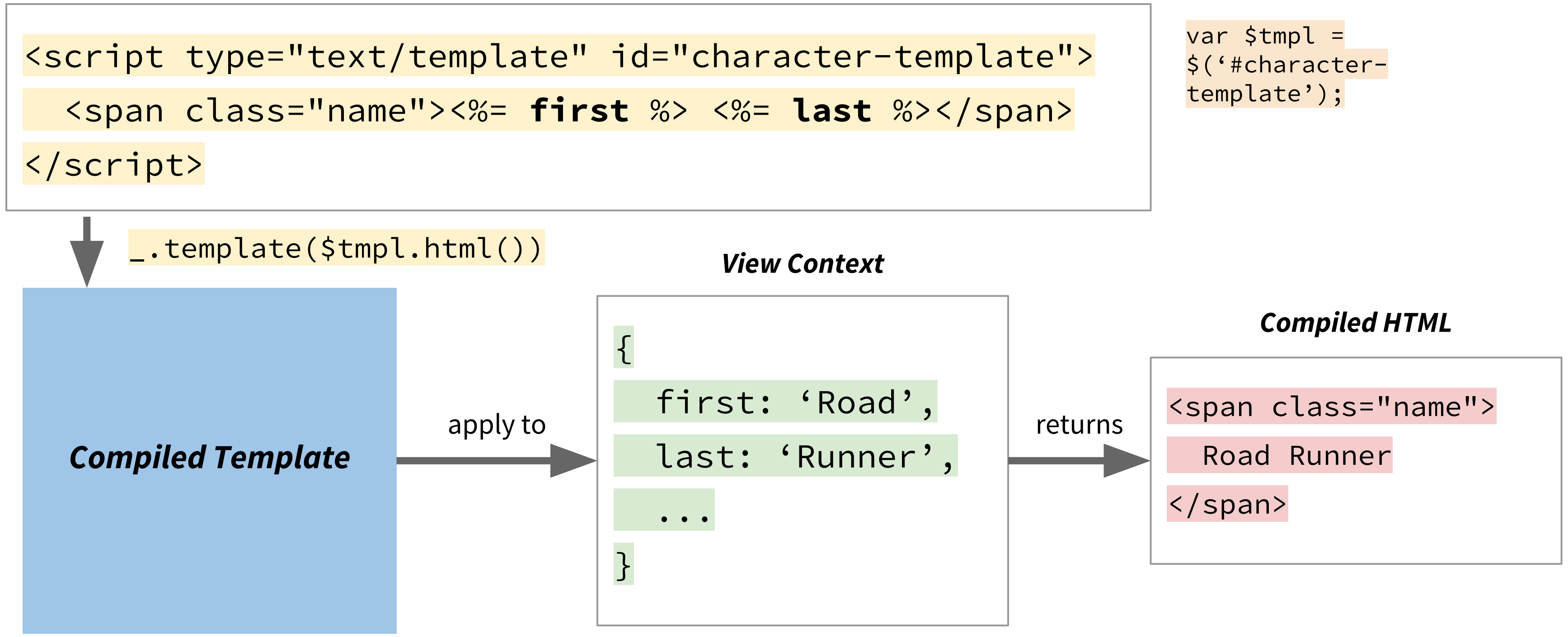
---

4. `render` invoked (returns `this` to allow *method chaining*)
5. place in DOM

To destroy:

1. unbind events
2. remove from DOM
3. (optional) destroy models

Underscore Templating



```
<script type="text/template" id="character-template">
  <span class="name"><%= first %> <%= last %></span>
</script>
```

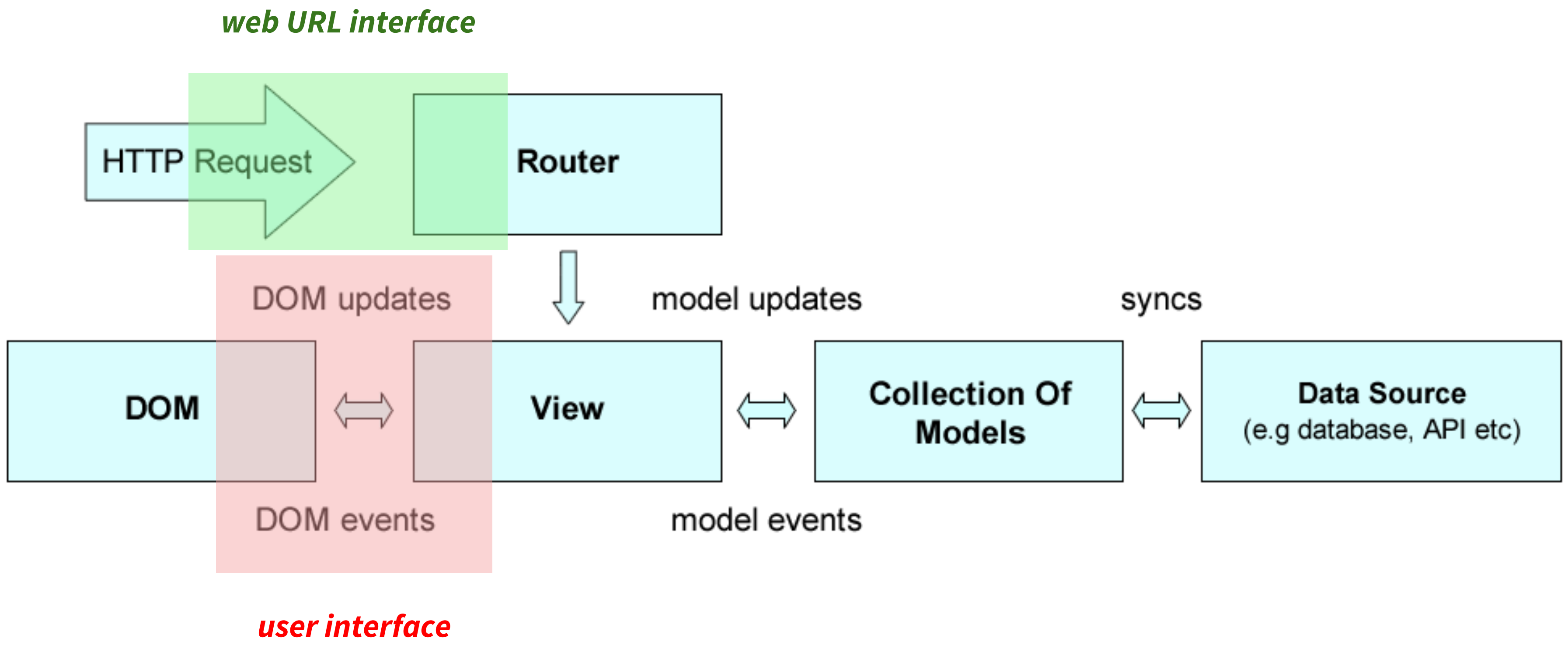
```
var $tmpl =
$('#character-
template');
```

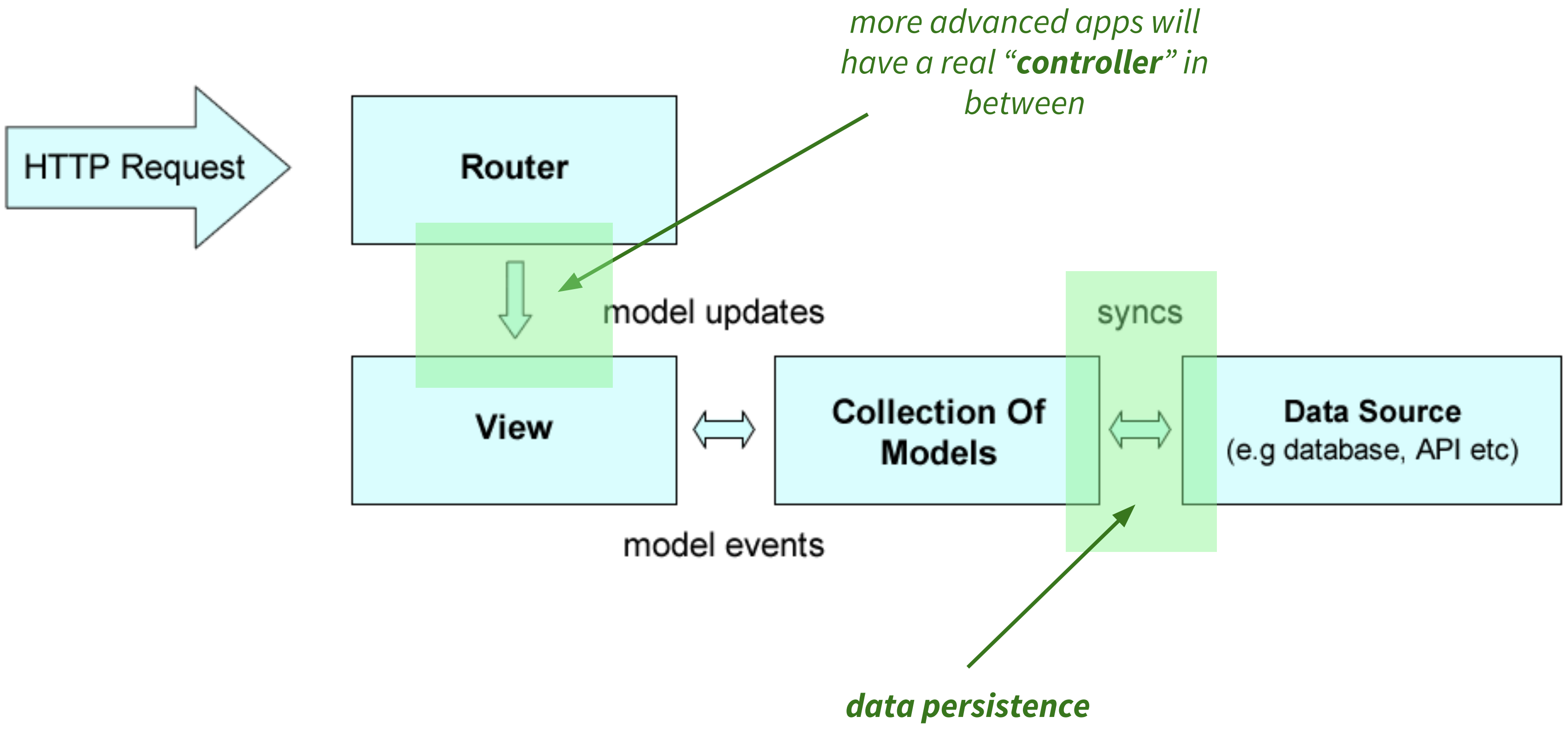
```
_.template($tmpl.html())
```

```
{
  first: 'Road',
  last: 'Runner',
  ...
}
```

```
<span class="name">
  Road Runner
</span>
```

ok, back to this picture!




more advanced apps will have a real "**controller**" in between
HTTP Request
Router
model updates
syncs
View
Collection Of Models
Data Source
(e.g database, API etc)
model events
data persistence

Intermediate stuff

(where Backbone really starts to shine)

Model persistence

- where can we store things?
- what is RESTful?
- `fetch`, `save`, `validate`

Router: connecting url fragments to parts of your application

- using `hashchange`
- using HTML5 `pushState`
- routing in practice

HW4

Due ***Sunday, 9 AM***