



# Palm® webOS™ 1-23

## Mojo, JavaScript & you

Davis W. Frank  
[dwfrank@pivotallabs.com](mailto:dwfrank@pivotallabs.com)  
<http://pivotallabs.com/users/dwfrank/blog>  
@dwfrank

Why?



Why?

**It's easy & familiar**

`my_app.is_a? Server`

`my_app.is_a? WebPage`

`myApp.constructor == document`

```
myApp = new Document("HTML5")
```

```
myApp = new Document("HTML5")
```

```
<script src="/usr/palm/frameworks/  
    mojo/mojo.js" type="text/  
javascript" x-mojo-version="1" />
```

**Mojo = API + MVC**

**Models are POJSO's**

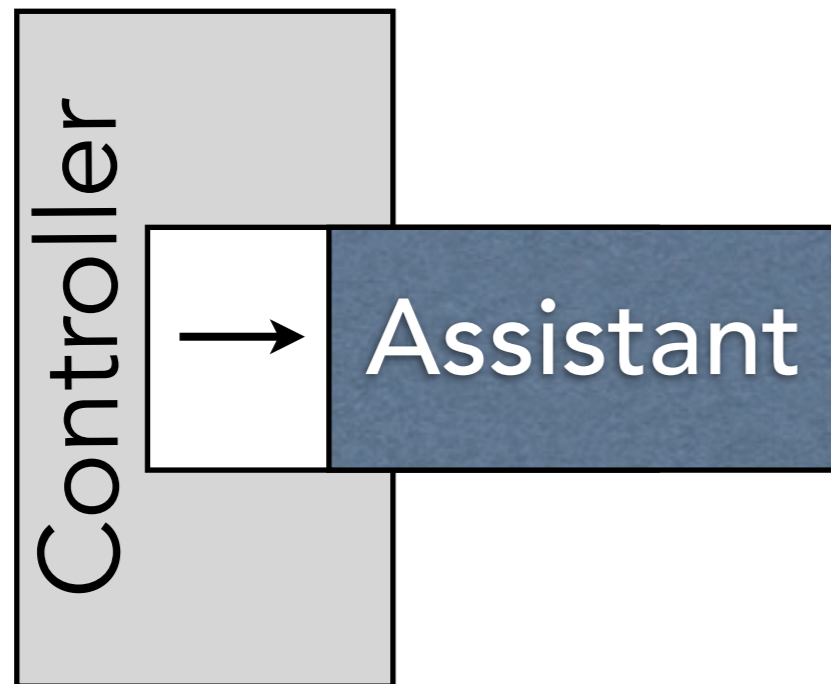
**Views are interpolated  
HTML files**

# Controllers are Assistants

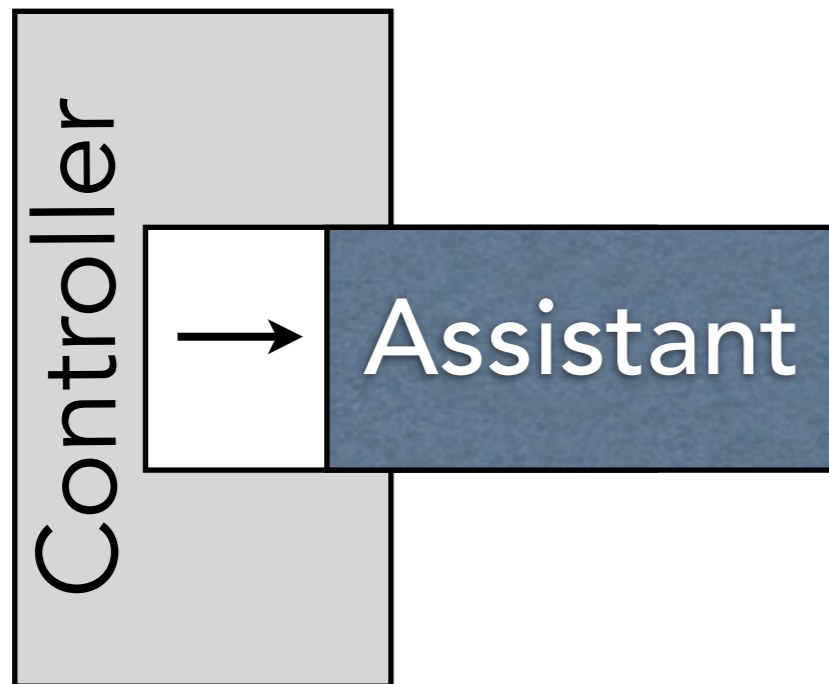
# Controllers are Assistants

kinda-sorta

# Assistant as Hybrid Template/Delegate

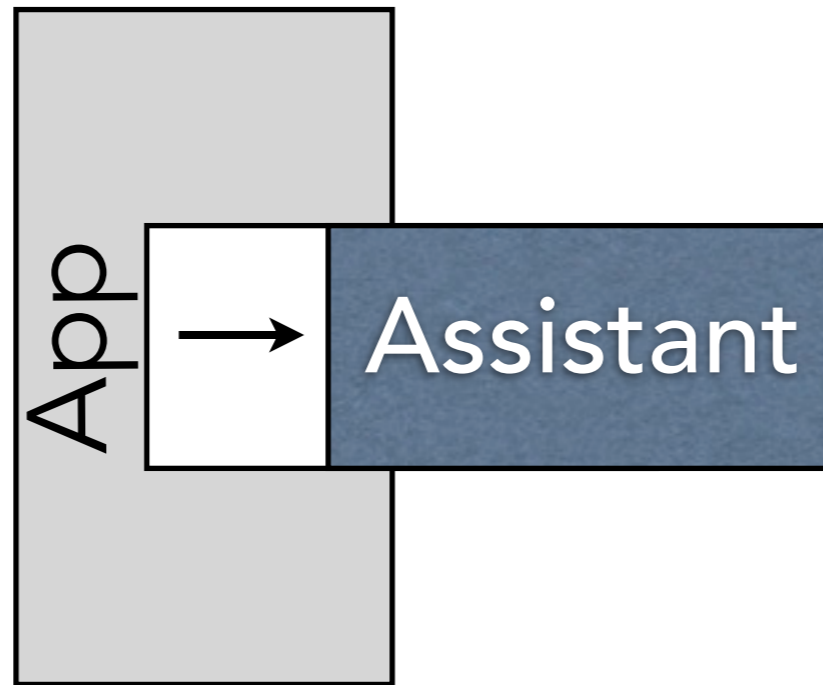


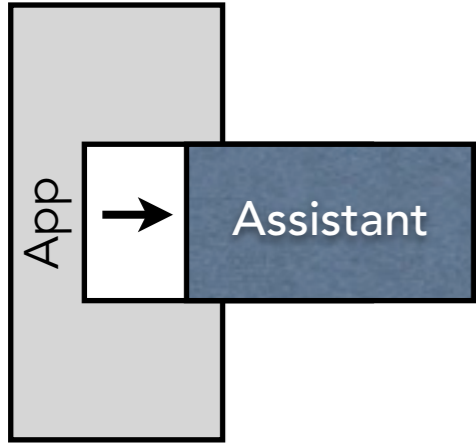
# Assistant as Hybrid Template/Delegate



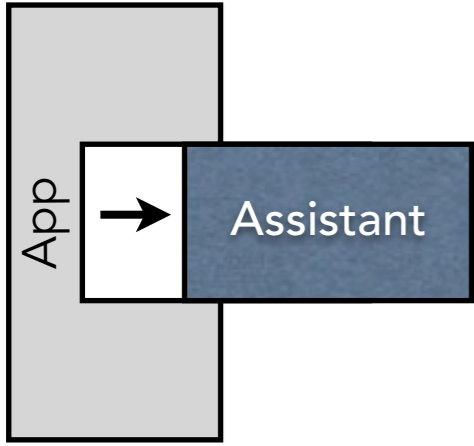
#setup  
#activate  
#deactivate  
#cleanup

# 3 C's

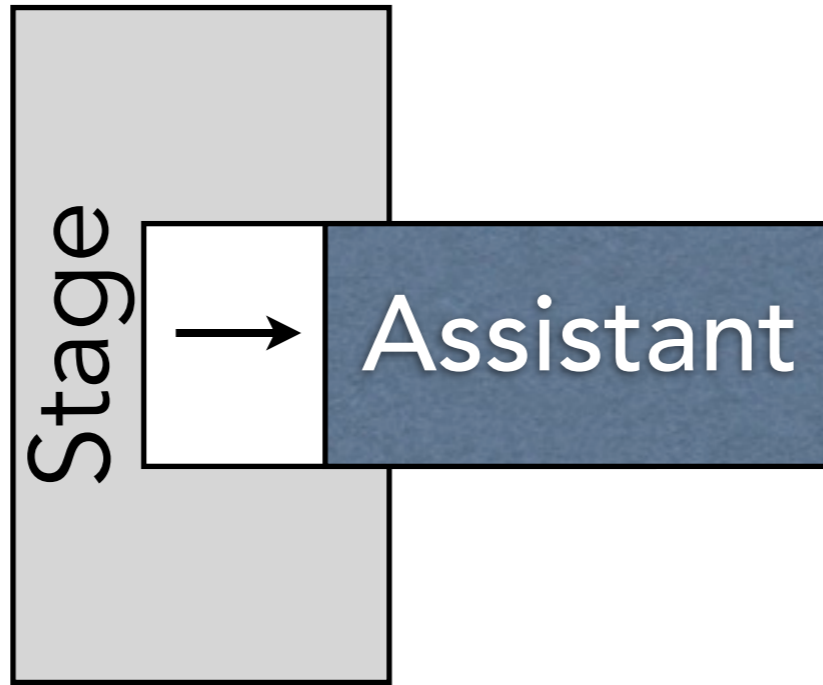




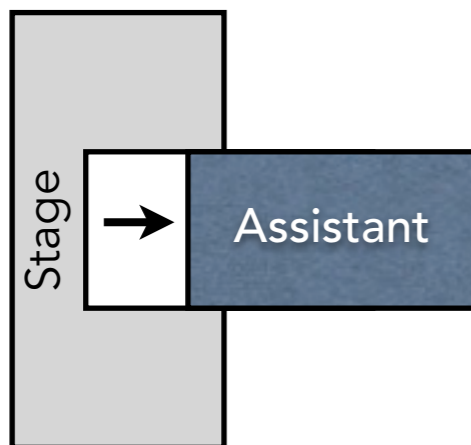
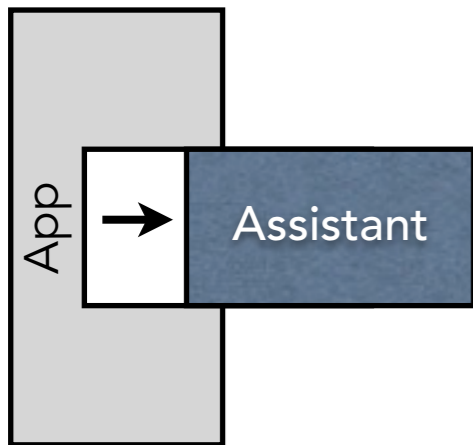
3 C's



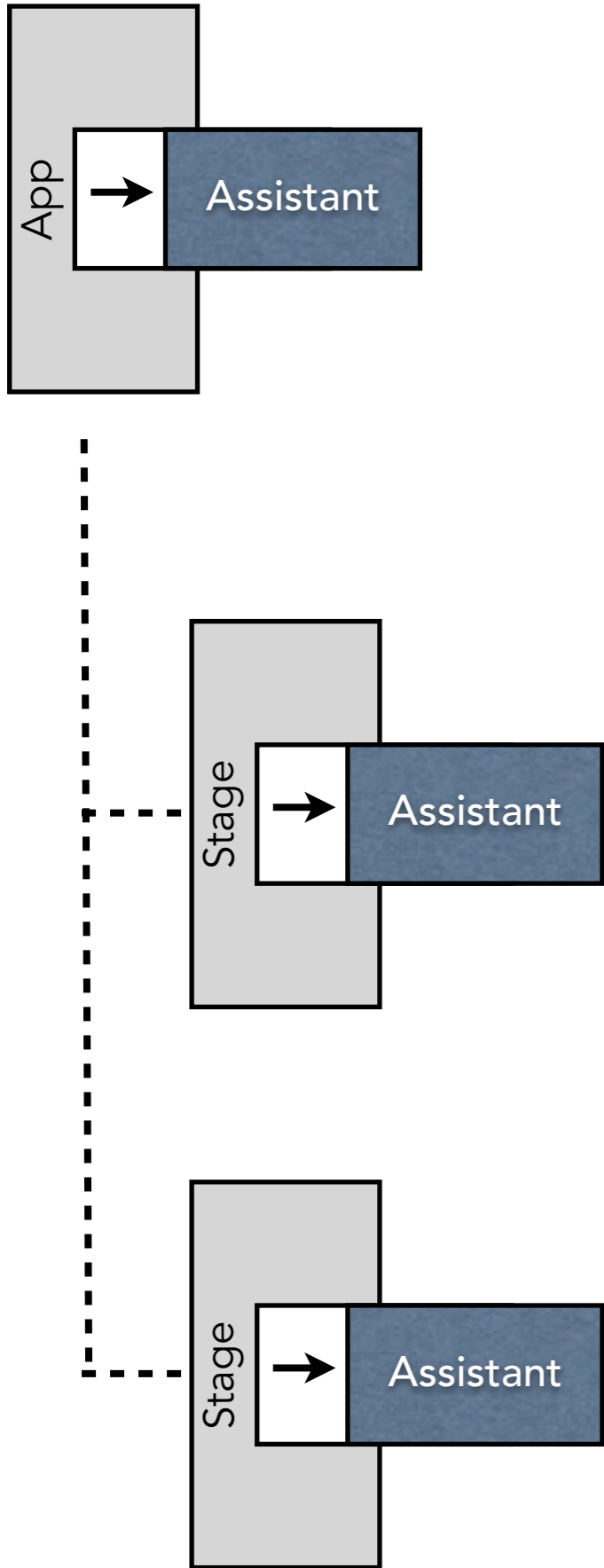
# 3 C's



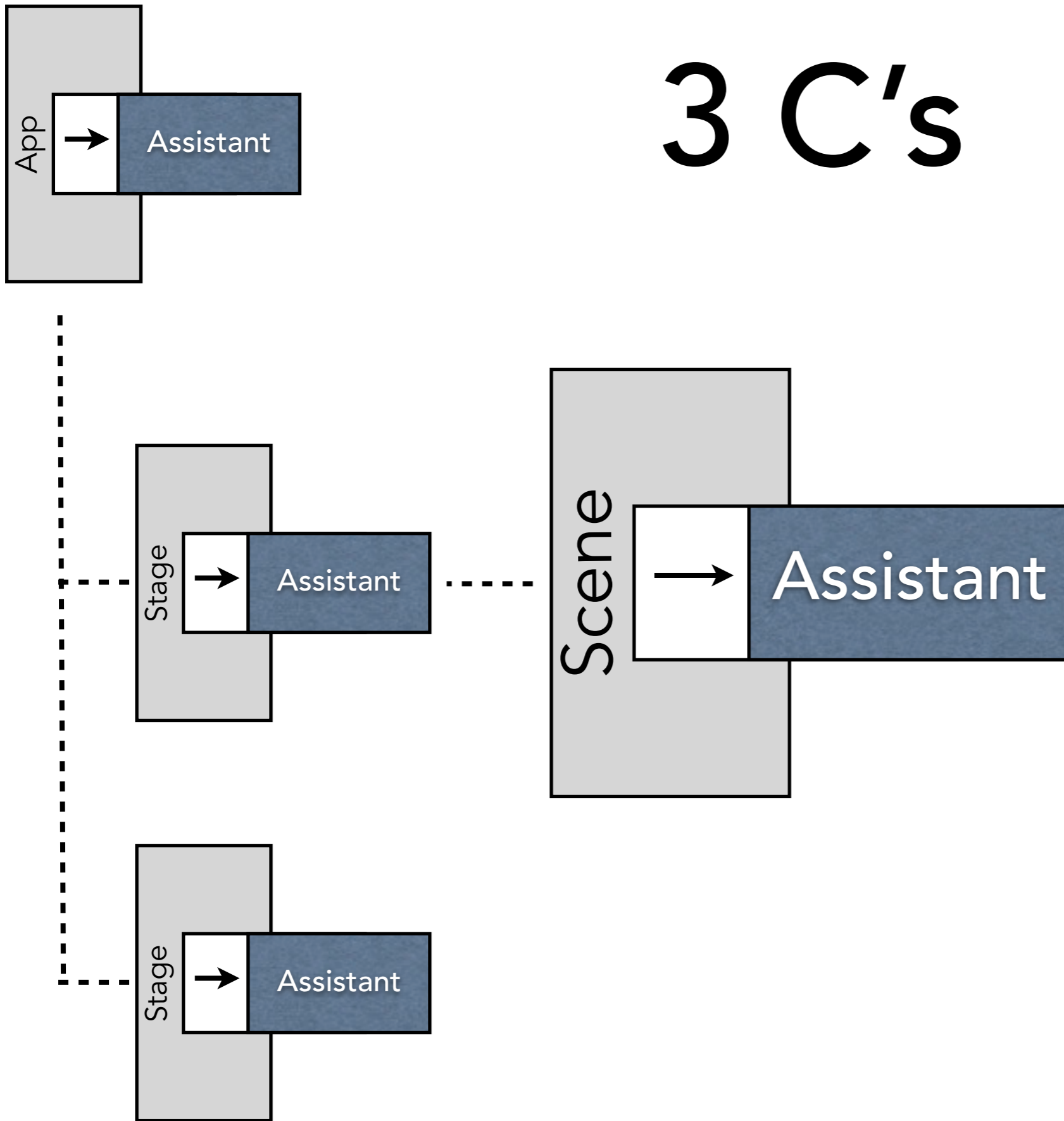
# 3 C's



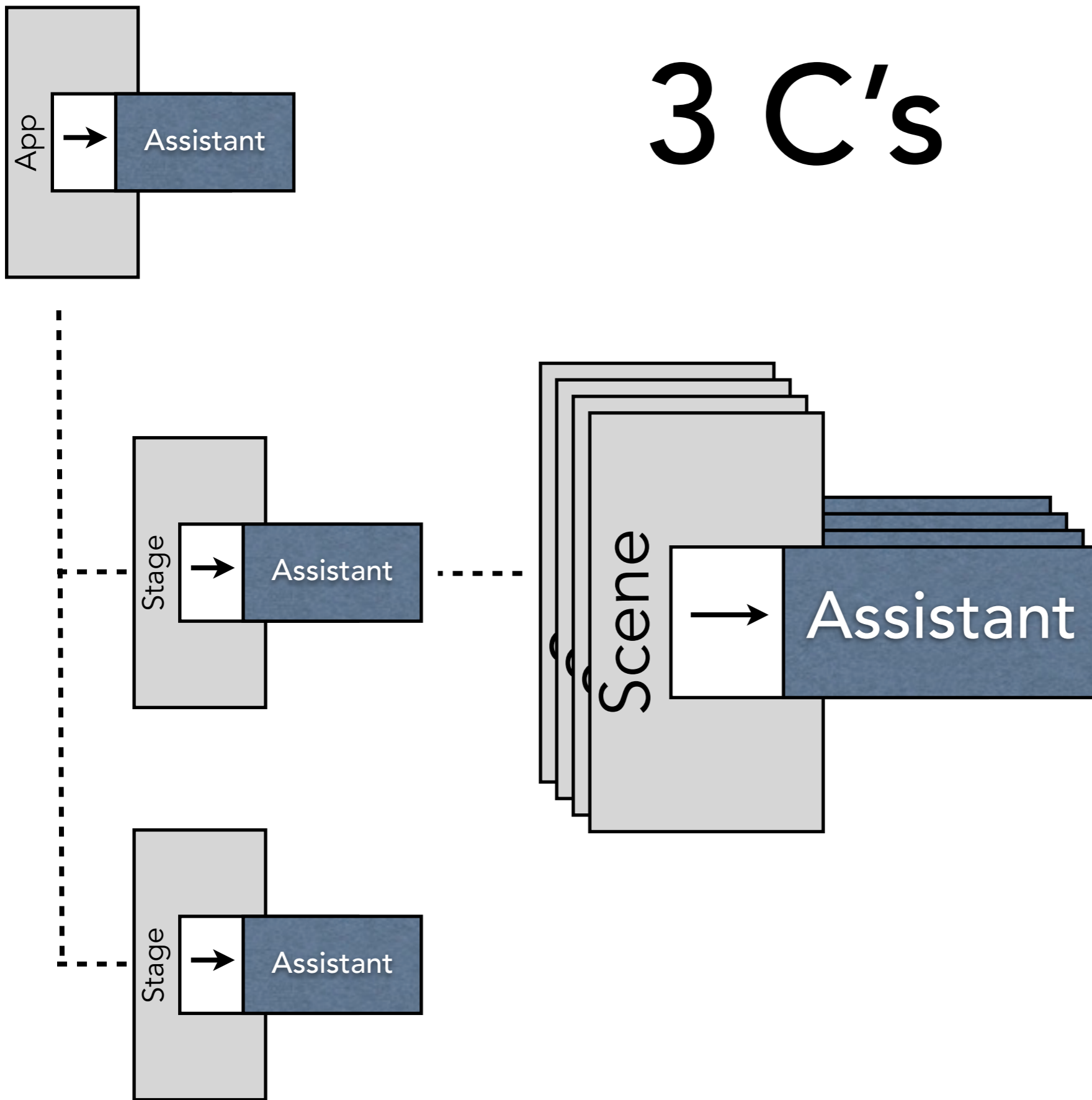
# 3 C's



# 3 C's



# 3 C's



**TATFT?**

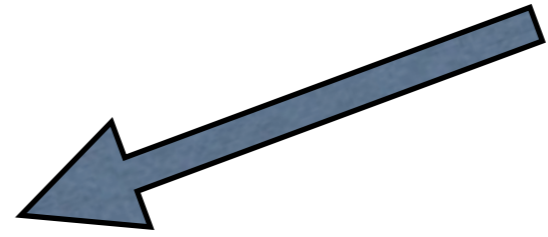
# Jasmine



```
// A Jasmine Spec
```

```
it("should be be able to make a deck of cards.", function()  
{  
    var deckOfCards = new DeckOfCards();  
    expect(deckOfCards.count).toEqual(52);  
});
```

```
// A Jasmine Suite
```



```
describe("DeckOfCards", function() {  
  
  it("should be be able to make a deck of cards.", function() {  
    var deckOfCards = new DeckOfCards();  
    expect(deckOfCards.count).toEqual(52);  
  });  
  
  it("should support shuffling", function() {  
    var deckOfCards = new DeckOfCards();  
    deckOfCards.shuffle();  
    expect(deckOfCards.count).toEqual(52);  
  });  
  
  it("should play 52 pickup", function() {  
    var deckOfCards = new DeckOfCards();  
    deckOfCards.play52Pickup();  
    expect(deckOfCards.count).toEqual(0);  
  });  
  
});
```

```
// A Jasmine Suite with a beforeEach
```

```
describe("DeckOfCards", function() {  
  var deckOfCards;
```

```
  beforeEach(function() {  
    deckOfCards = new DeckOfCards();  
  });
```

```
  it("should be able to make a deck of cards.", function() {  
    expect(deckOfCards.count).toEqual(52);  
  });
```

```
  it("should support shuffling", function() {  
    deckOfCards.shuffle();  
    expect(deckOfCards.count).toEqual(52);  
  });
```

```
  it("should play 52 pickup", function() {  
    deckOfCards.play52Pickup();  
    expect(deckOfCards.count).toEqual(0);  
  });
```

```
});
```

```
// A Jasmine Suite with nested describes
```

```
describe("DeckOfCards", function() {  
  var deckOfCards;
```

```
  beforeEach(function() {  
    deckOfCards = new DeckOfCards();  
  });
```

```
  it("should be be able to make a deck of cards.", function() {  
    expect(deckOfCards.count).toEqual(52);  
  });
```

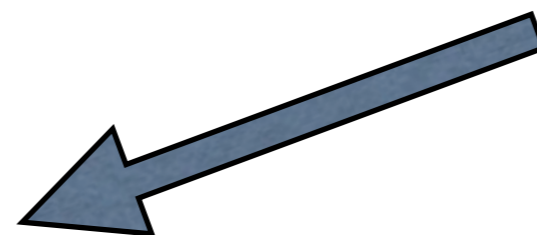
```
...
```

```
describe("with Jokers", function() {  
  deckWithJokers = new DeckOfCards({jokers: true});
```

```
  it("should support shuffling without losing any cards", function() {  
    deckWithJokers.shuffle();  
    expect(deckWithJokers.count).toEqual(54);  
  });
```

```
  it("should play 52 pickup", function() {  
    deckWithJokers.play52Pickup();  
    expect(deckWithJokers.count).toEqual(0);  
  });
```

```
});  
});
```



```
// Using a Jasmine Spy
```

```
describe("BlackJackDealer", function() {
```

```
  var dealer, pitBoss;
```

```
  beforeEach(function() {
```

```
    dealer = new BlackJackDealer();
```

```
    pitBoss = {getNewBlackJackDeck: function() {}};
```

```
    spyOn(pitBoss, 'getNewBlackJackDeck');
```

```
  });
```

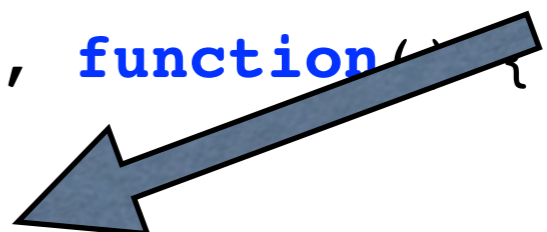
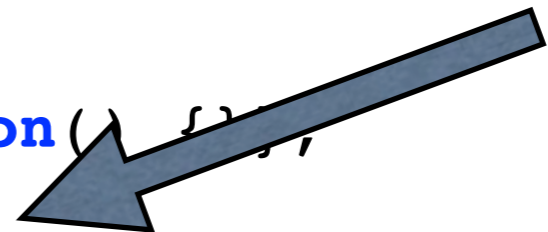
```
  it("should get a new deck from the pit boss", function() {
```

```
    dealer.refreshDeck();
```

```
    expect(pitBoss.getNewBlackJackDeck).toHaveBeenCalled();
```

```
  });
```

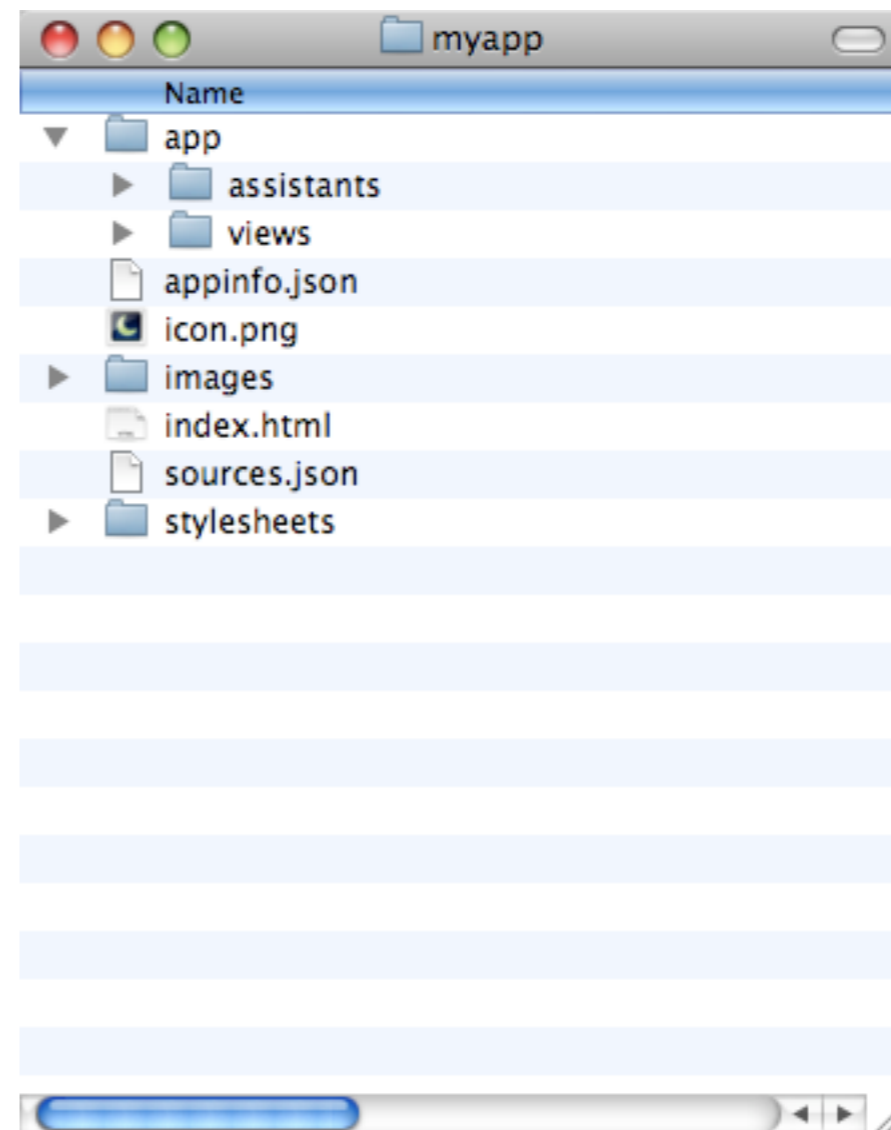
```
});
```



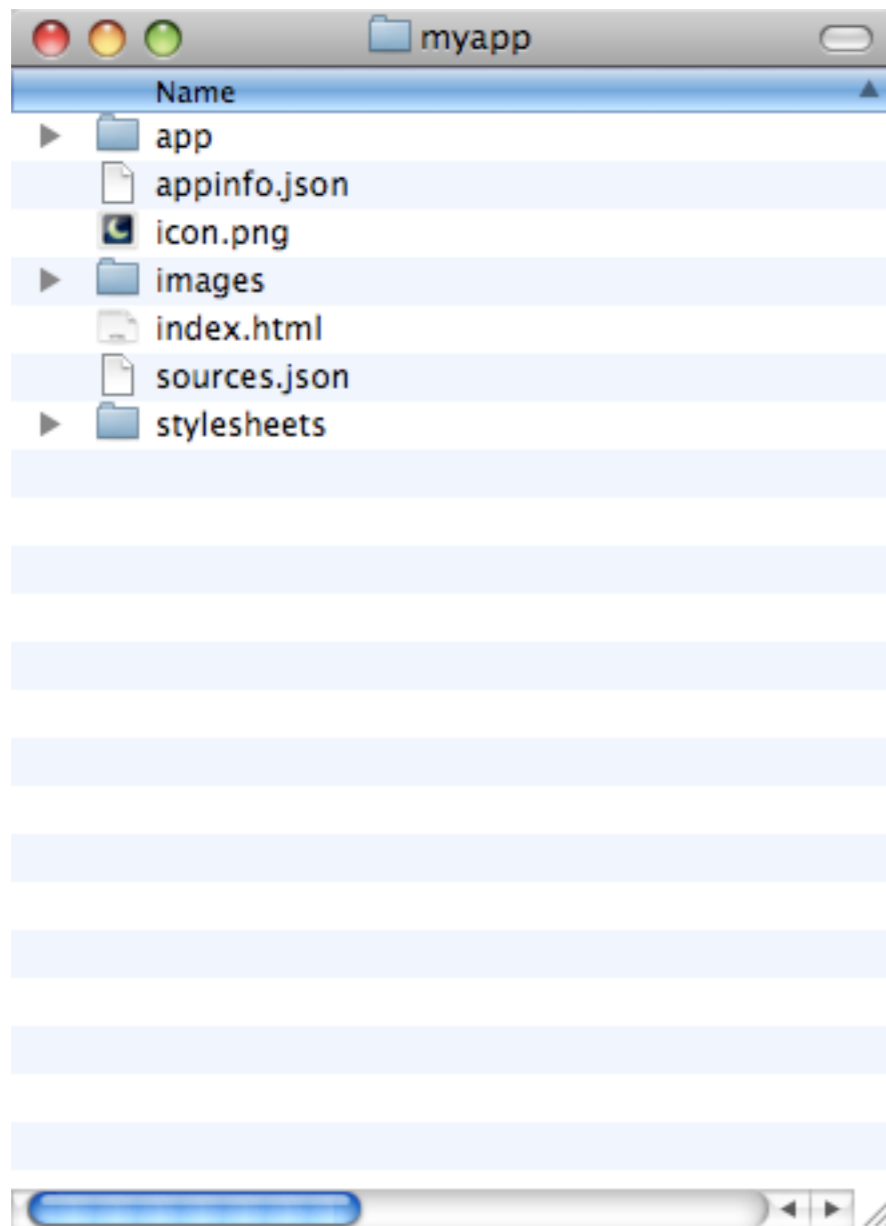
# Jasmine on webOS?

# This is your App

```
$ palm-generate myapp
```

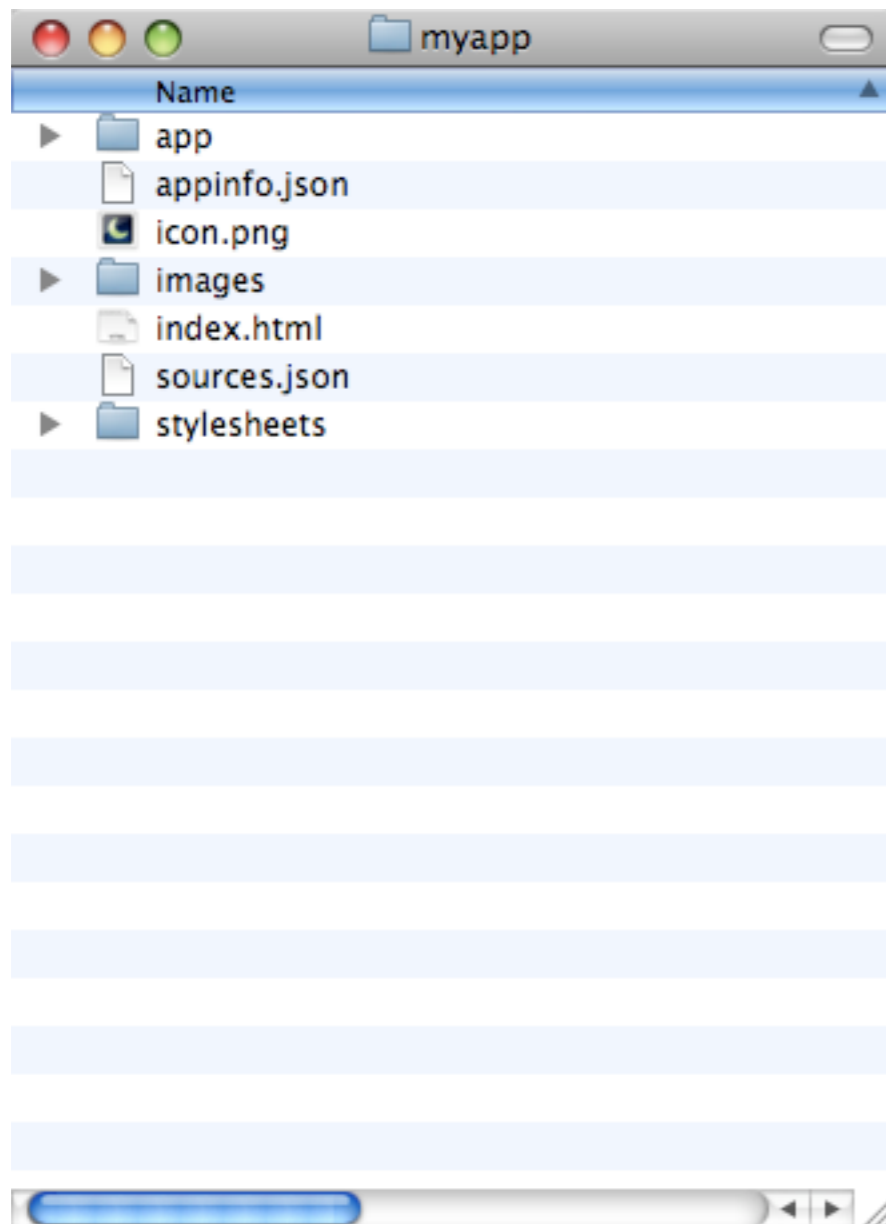


# ...in Pockets



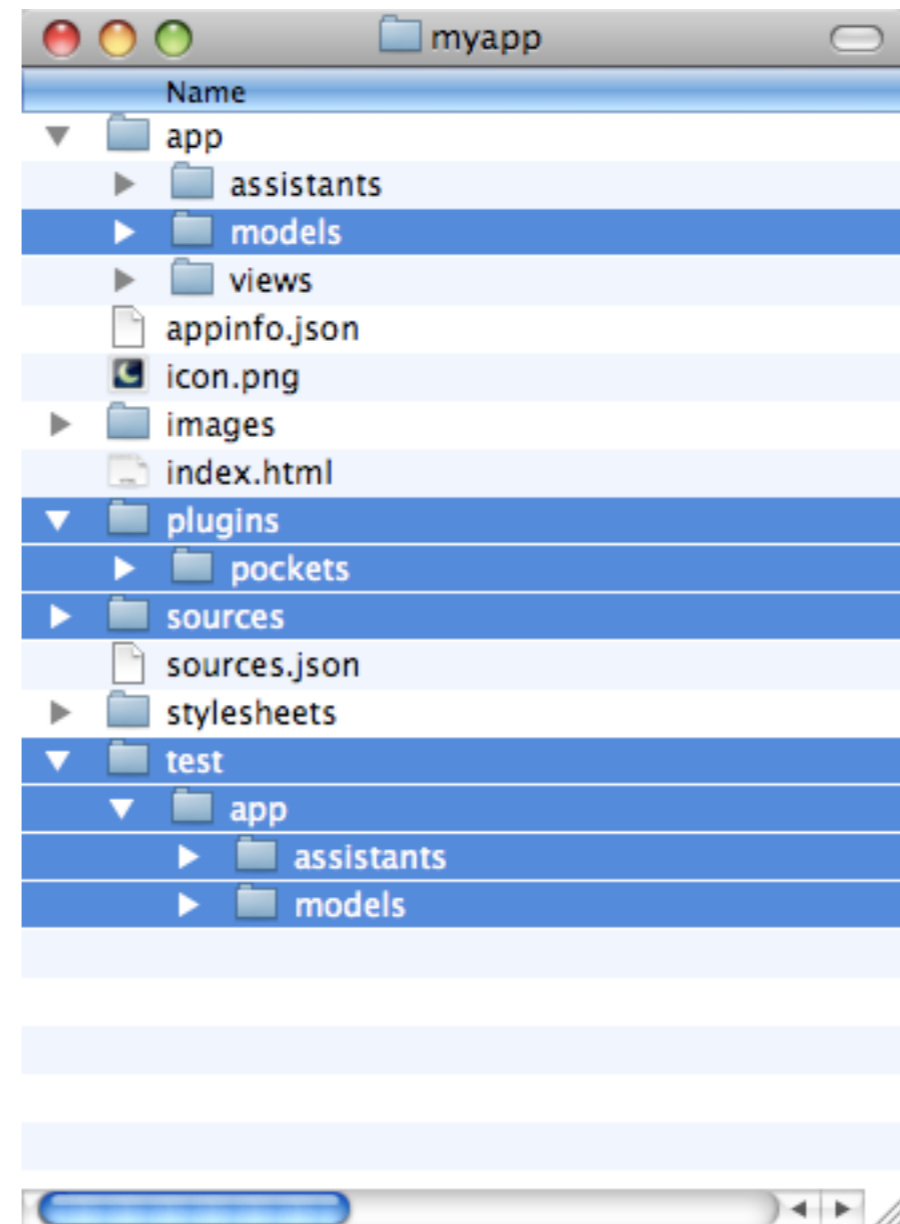
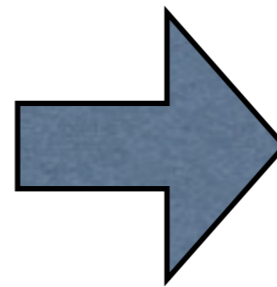
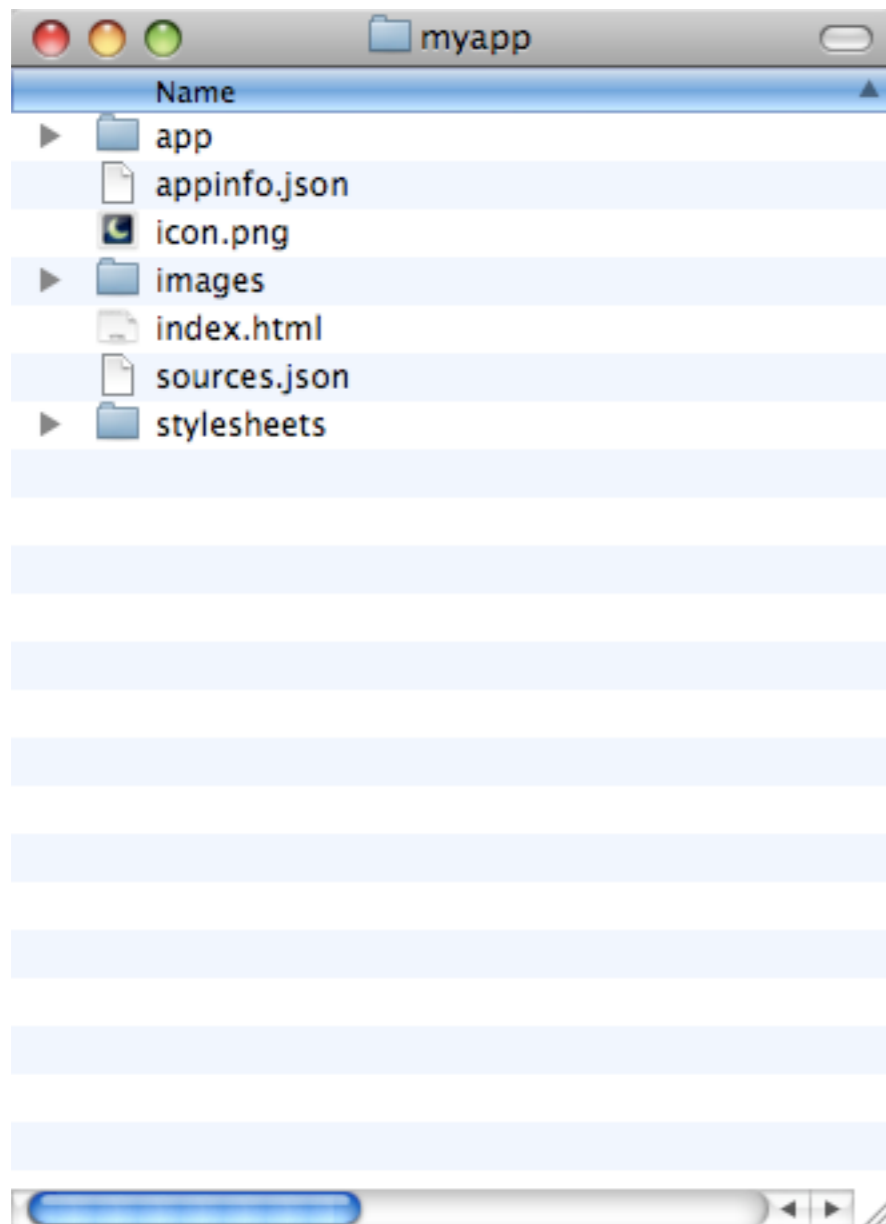
# ...in Pockets

```
$ pockets generate myapp
```

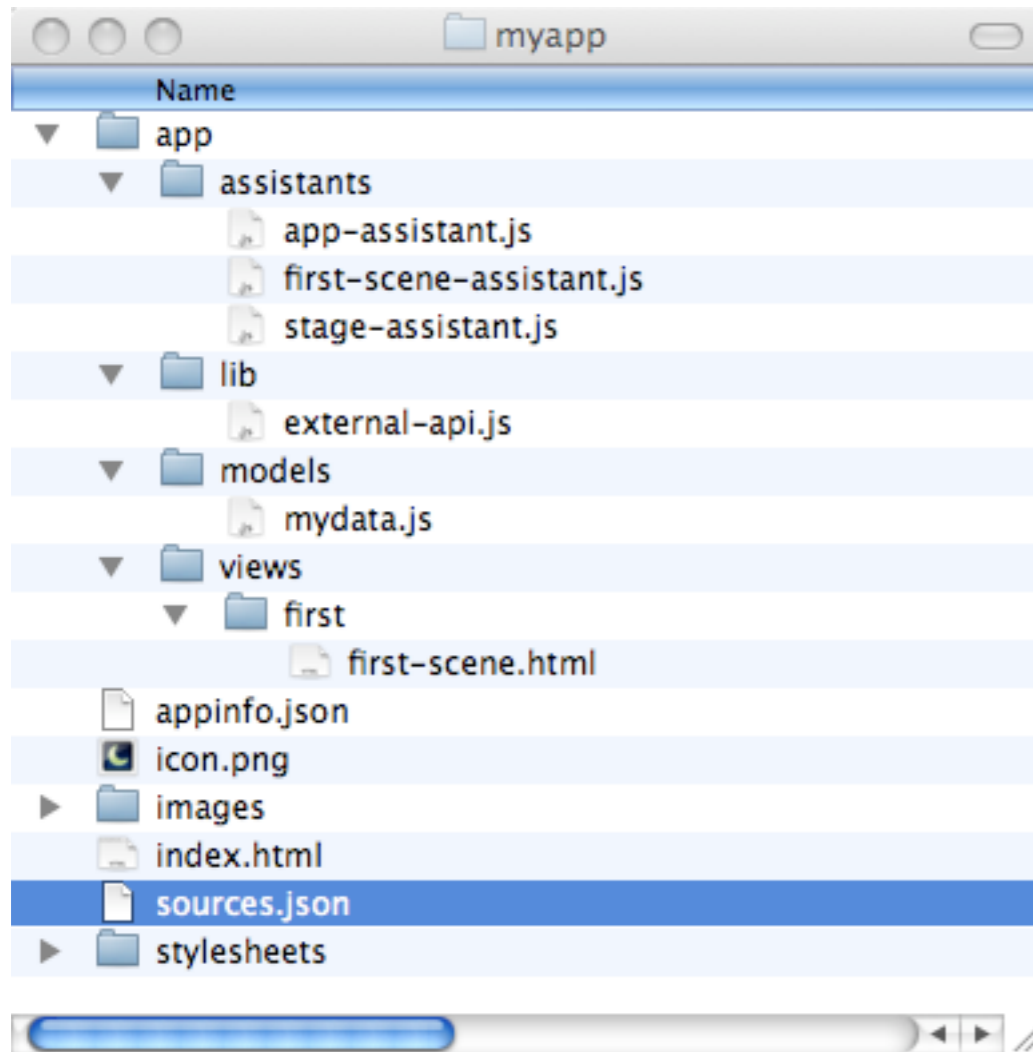


# ...in Pockets

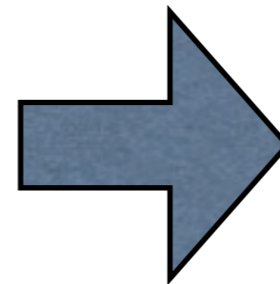
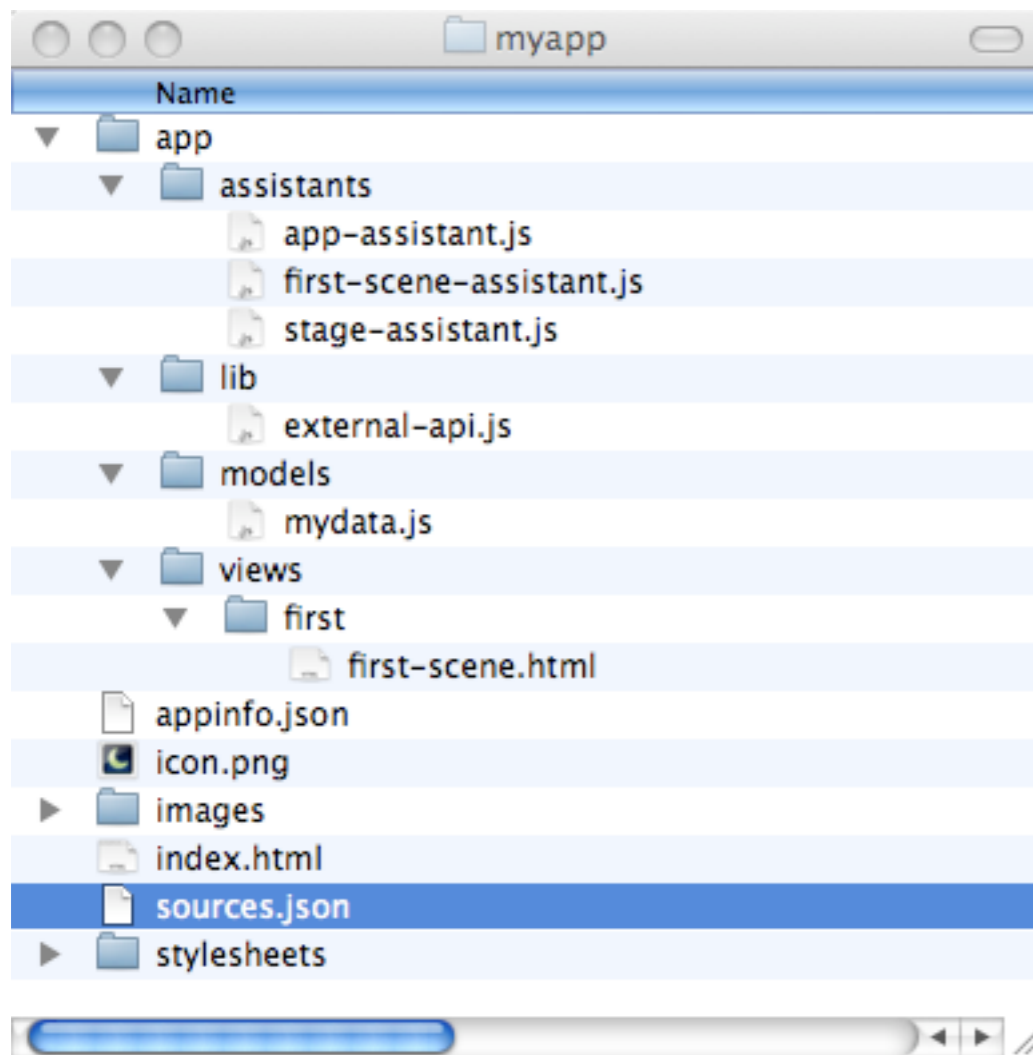
```
$ pockets generate myapp
```



# sources.json



# sources.json



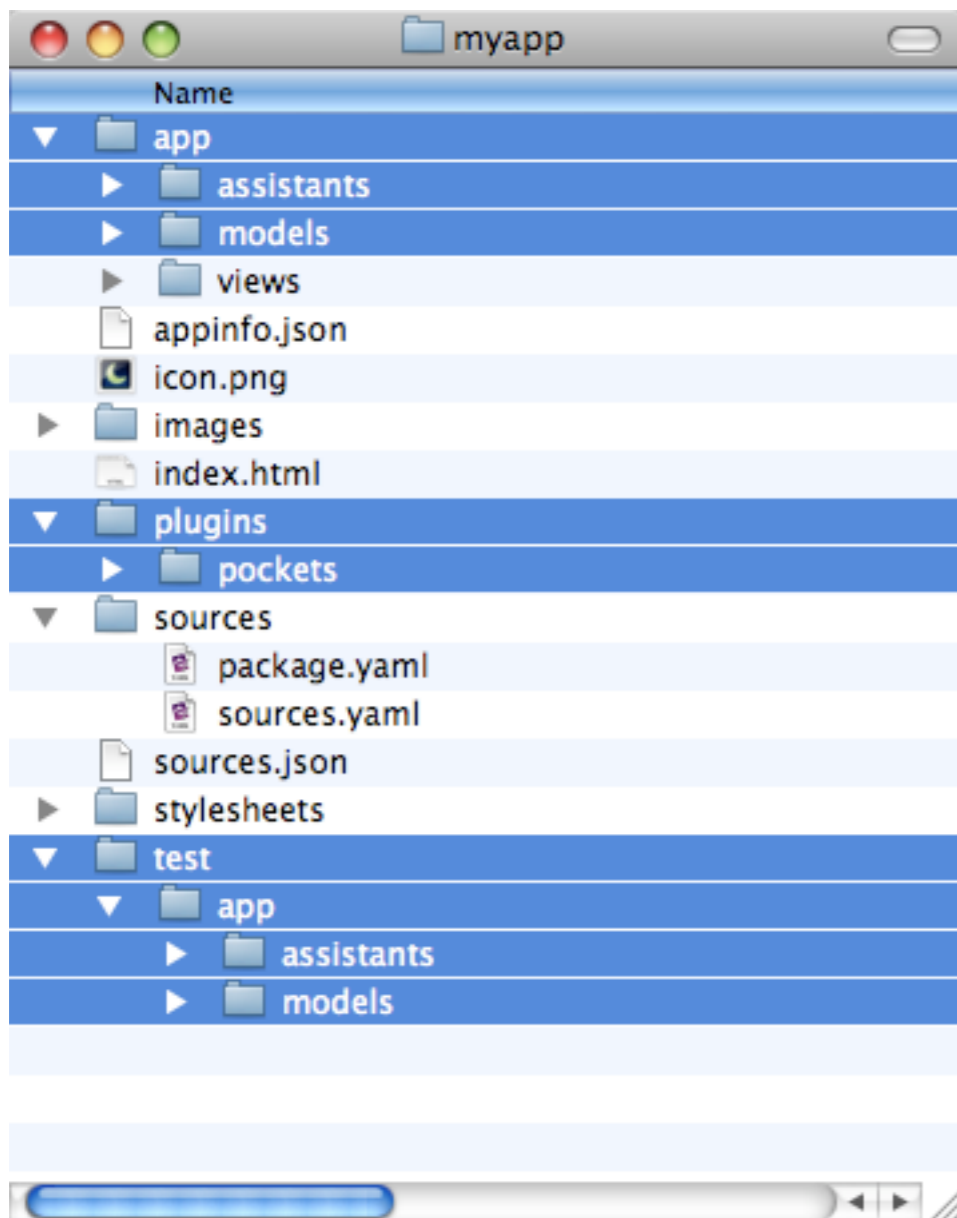
```
[{"source": "vendor\\pockets\\lib\\production\\pockets.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-object.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-collections.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-managers.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-observables.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-subscribables.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-event-target.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-ui.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-ui-base-scene-assistant.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-dom.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-monkeypatch.js"}, {"source": "vendor\\pockets\\lib\\production\\pockets-gestures.js"}, {"source": "app\\lib\\date-helper.js"}, {"source": "app\\lib\\formatters.js"}, {"source": "app\\models\\tweet-results.js"}, {"source": "app\\assistants\\base-scene-assistant.js"}, {"source": "app\\assistants\\timeline-assistant.js"}, {"source": "app\\assistants\\message-timeline-assistant.js"}, {"source": "app\\assistants\\status-timeline-assistant.js"}, {"source": "app\\assistants\\users-timeline-assistant.js"}, {"source": "app\\assistants\\user-timeline-assistant.js"}, {"source": "app\\assistants\\base-tweet-assistant.js"}, {"source": "lib\\webtoolkit.base64.js"}, {"source": "lib\\support-info.js"}, {"source": "app\\lib\\tweet-view.js"}, {"source": "app\\lib\\card-stack.js"}, {"source": "app\\lib\\depot-manager.js"}, {"source": "app\\lib\\error-prompter.js"}, {"source": "app\\lib\\initializer.js"}, {"source": "app\\lib\\message-view.js"}, {"source": "app\\lib\\rate-limit.js"}, {"source": "app\\lib\\stage-manager.js"}, {"source": "app\\lib\\tinyurl-api.js"}, {"source": "app\\lib\\trend-list.js"}, {"source": "app\\lib\\tweet-link-registry.js"}, {"source": "app\\lib\\twitter-api.js"}, {"source": "app\\models\\account.js"}, {"source": "app\\models\\accounts.js"}, {"source": "app\\models\\bookmark.js"}, {"source": "app\\models\\bookmarks.js"}, {"source": "app\\models\\direct-message.js"}, {"source": "app\\models\\direct-messages-received.js"}, {"source": "app\\models\\direct-messages-sent.js"}, {"source": "app\\models\\favorites-results.js"}, {"source": "app\\models\\followers.js"}, {"source": "app\\models\\following.js"}, {"source": "app\\models\\home-timeline.js"}, {"source": "app\\models\\preferences.js"}, {"source": "app\\models\\public-timeline-results.js"}, {"source": "app\\models\\replies.js"}, {"source": "app\\models\\search-results.js"}, {"source": "app\\models\\trend.js"}, {"source": "app\\models\\tweet.js"}, {"source": "app\\models\\twitter-error.js"}, {"source": "app\\models\\unauthenticated-account.js"}, {"source": "app\\models\\user-profile.js"}, {"source": "app\\models\\user-timeline-results.js"}, {"source": "app\\models\\user-timeline-results.js"}]
```

# sources.json

```
$ pockets sources [--production]
```

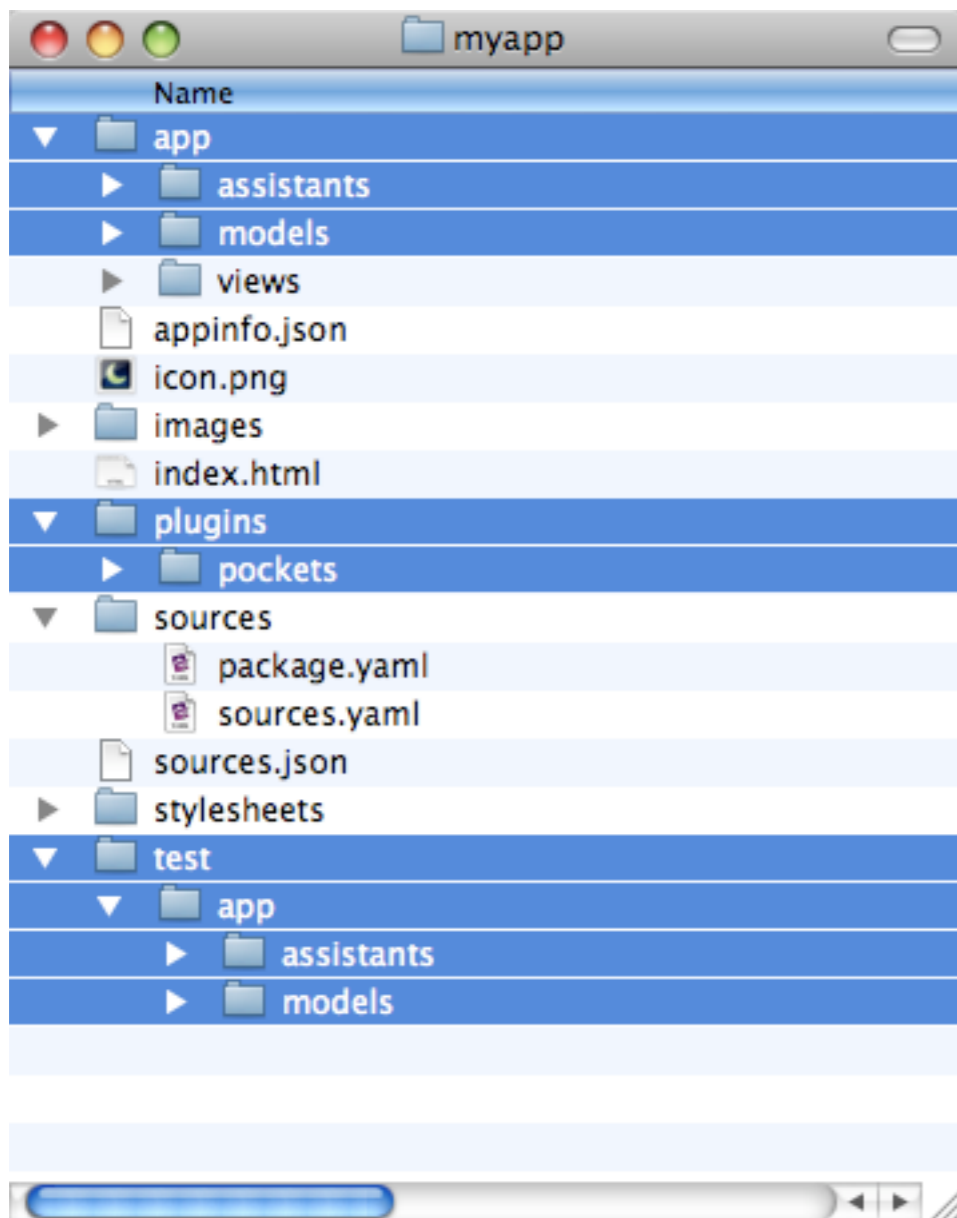
# sources.json

```
$ pockets sources [--production]
```



# sources.json

```
$ pockets sources [--production]
```



Icons by [DryIcons](#)

**Wanna play?**  
***Let's play.***

# Tips & Tricks

# JS isn't afraid of Objects & Patterns

**API's deserve objects**

# Skinny Controller, Fat Model

# Skinnny ~~Controller,~~ Fat Model

# Skinny Assistant, Fat Model

Use your  
`this.controller.sceneElement`

a word about  
Prototype.js

a word about  
Prototype.js

meh.

```
this.controller.get  
this.controller.select
```

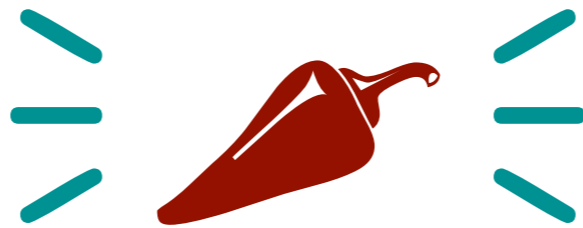
~~this.controller.get~~  
~~this.controller.select~~

sceneElement.querySelector  
sceneElement.querySelectorAll

<XML> + 

<XML> + 

===



# Two Words: One Browser

**make a CSScene**

**Always add class**

**Use the (Mojo) source**

THX,

DWF

**@pivotalpockets**