

Composition

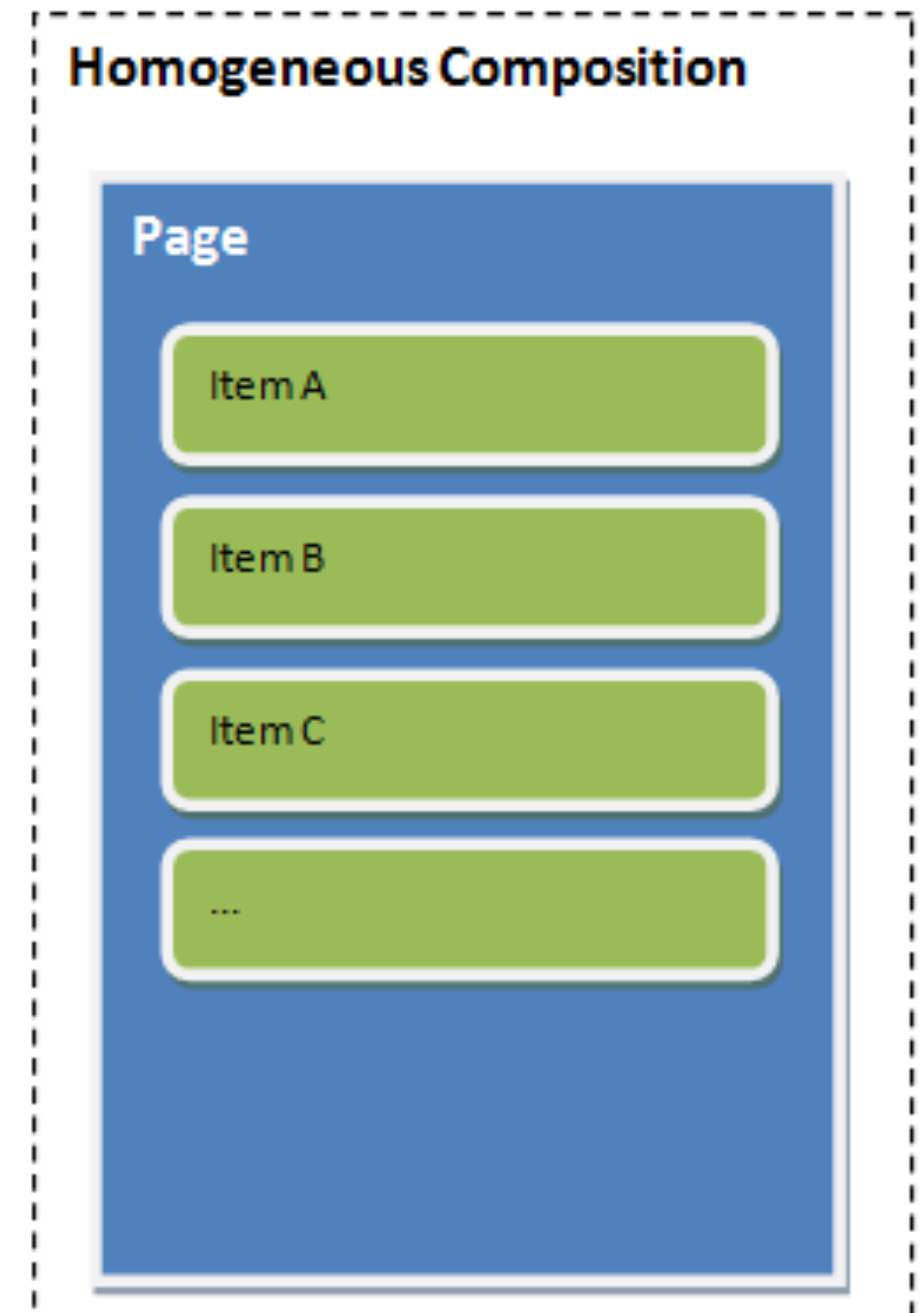
Composition

- Assemble complex screens out of small reusable components
- Aggregate multiple distinct sub-views into a more complex view

<https://www.sitepoint.com/composition-aurelia-report-builder/>

Homogenous Composition

- Rendering items which have the same type and only varying content.
- This type of composition is used in most frameworks when creating repeated lists.
- .. a simple list of items being rendered sequentially one after another.



Donations to Date

Amount	Method donated	Candidate
23	cash	Simpson, Lisa
212	paypal	Simpson, Bart
3	Cash	Simpson, Lisa
3	Cash	Simpson, Lisa
3	Cash	Simpson, Lisa

<template>

```

<article class="ui stacked segment">
  <h3 class='ui dividing header'> Donations to Date </h3>
  <table class="ui celled table segment">
    <thead>
      <tr>
        <th>Amount</th>
        <th>Method donated</th>
        <th>Candidate</th>
      </tr>
    </thead>
    <tbody>
      <tr repeat.for="donation of donations">
        <td> ${donation.amount}</td>
        <td> ${donation.method}</td>
        <td> ${donation.candidate.lastName},
          ${donation.candidate.firstName}</td>
      </tr>
    </tbody>
  </table>
</article>

```

</template>

Homogenous Composition Example

```

import {inject} from 'aurelia-framework';
import DonationService from '../services/donation-service';

@inject(DonationService)
export class Report {

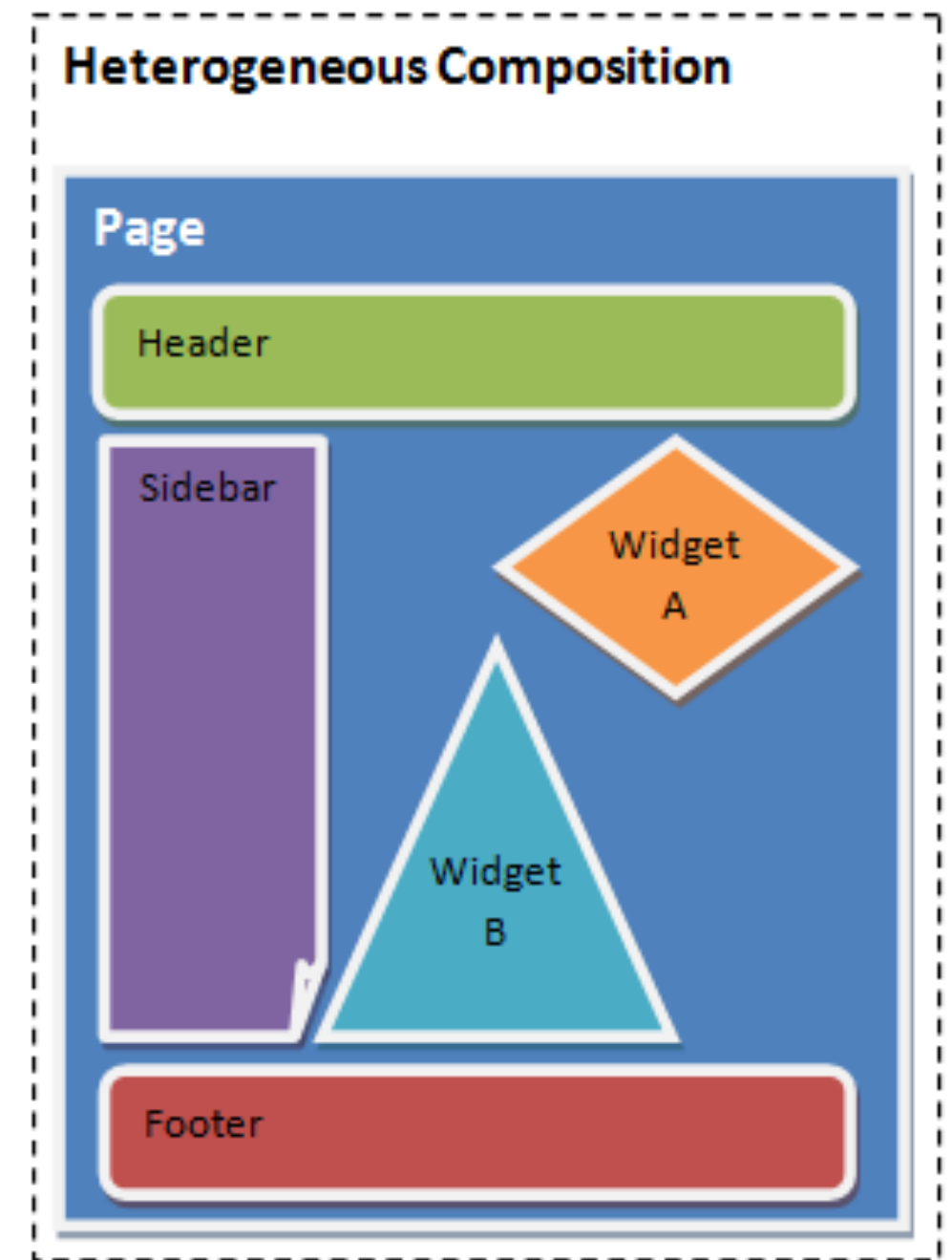
  donations = [];

  constructor(ds) {
    this.donationService = ds;
    this.donations = this.donationService.donations;
  }
}

```

Heterogeneous Composition

- Assembly of items which have different types and views
- Encourage developer towards the creation of small and reusable components
- In order to adhere to the DRY Principle, we don't necessarily want to rely on tight coupling between our view and view-model pairs.
- Using `<compose>` we can do this in aurelia



- Three View/View-Model Pairs
- Donation
- Report
- Stats

Make a Donation

Amount

Select Method

☒ Cash
☐ PayPal

Cash

Select Candidate

☒ Simpson, Lisa
☐ Simpson, Bart

Lisa Simpson

Donate

Donations to Date

Amount	Method donated	Candidate
23	cash	Simpson, Lisa
212	paypal	Simpson, Bart

0
DONATED

Compose Element in app view/view-model

```
<section class="ui two column stackable grid basic segment">
  <aside class="column">
    <compose view-model="./viewmodels/donate"></compose>
  </aside>
  <article class="column">
    <compose view-model="./viewmodels/report"></compose>
  </article>
  <article class="column">
    <compose view-model="./viewmodels/stats"></compose>
  </article>
</div>
</section>
```

```
export class App {
}
```

Make a Donation

Amount

Select Method

☒ Cash
☐ PayPal

Cash

Select Candidate

☒ Simpson, Lisa
☐ Simpson, Bart

Lisa Simpson

Donate

Donations to Date

Amount	Method donated	Candidate
23	cash	Simpson, Lisa
212	paypal	Simpson, Bart

0

DONATED

Binding

- A view, with now explicit view-model
- It still expects to bind to 'login()', 'email' and 'password'.

```
<template>
  <form submit.delegate="login($event)" class="ui stacked segment form">
    <h3 class="ui header">Log-in</h3>
    <div class="field">
      <label>Email</label> <input placeholder="Email" value.bind="email"/>
    </div>
    <div class="field">
      <label>Password</label> <input type="password" value.bind="password"/>
    </div>
    <button class="ui blue submit button">Login</button>
    <h3>${prompt}</h3>
  </form>
</template>
```

Log-in

Email

marge@simpson.com

Password

•••••

Login

Binding login view

app.html

```
<template>
  <div class="ui container">
    <section class="ui four column stackable grid basic segment">
      <div class="ui row">
        <section class="ui five wide column">
          <compose view="./viewmodels/login.html"></compose>
        </section>
      </div>
      <div class="ui row">
        <aside class="column">
          <compose view-model="./viewmodels/donate"></compose>
        </aside>
        <article class="column">
          <compose view-model="./viewmodels/report"></compose>
        </article>
        <article class="column">
          <compose view-model="./viewmodels/candidates"></compose>
        </article>
        <article class="column">
          <compose view-model="./viewmodels/stats"></compose>
        </article>
      </div>
    </section>
  </div>
</template>
```

Log-in

Email

Password

Login

Make a Donation

Amount

Select Method

☒ Cash
☐ PayPal

Cash

Select Candidate

☒ Simpson, Lisa
☐ Simpson, Bart

Lisa Simpson

Donate

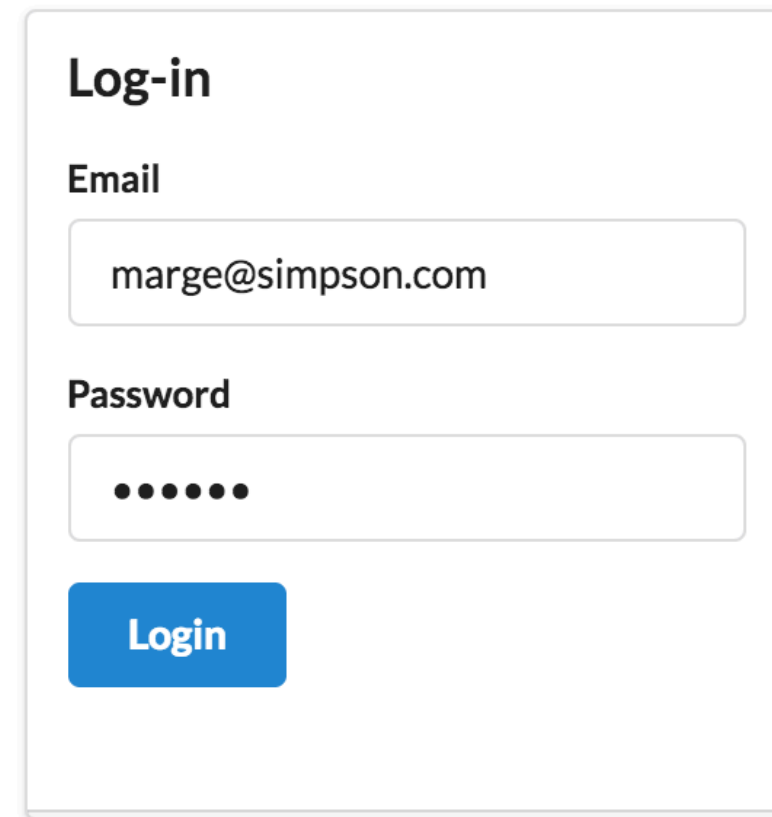
Donations to Date

Amount	Method donated	Candidate
23	cash	Simpson, Lisa
212	paypal	Simpson, Bart

0
DONATED

- login composed as a 'view', not a 'view-model'

Binding login view



Log-in

Email

marge@simpson.com

Password

••••••

Login

app.js

```
export class App {  
  
  email = 'marge@simpson.com';  
  password = 'secret';  
  
  loggedIn = false;  
  
  login() {  
    console.log(`Logging in ${this.email}`);  
    this.loggedIn = true;  
  }  
}
```

login.html

```
<template>  
  
  <form submit.delegate="login($event)" class="ui stacked segment form">  
    <h3 class="ui header">Log-in</h3>  
    <div class="field">  
      <label>Email</label> <input placeholder="Email" value.bind="email"/>  
    </div>  
    <div class="field">  
      <label>Password</label> <input type="password" value.bind="password"/>  
    </div>  
    <button class="ui blue submit button">Login</button>  
    <h3>${prompt}</h3>  
  </form>  
  
</template>
```

- login composed as a 'view', not a 'view-model'
- ... and will be bound to app.js

Selective Display

```
<template>
  <div class="ui container">
    <section class="ui four column stackable grid basic segment">
      <div show.bind="!loggedIn" class="ui row">
        <section class="ui five wide column">
          <compose view="./viewmodels/login.html"></compose>
        </section>
      </div>
      <div show.bind="loggedIn" class="ui row">
        <aside class="column">
          <compose view-model="./viewmodels/donate"></compose>
        </aside>
        <article class="column">
          <compose view-model="./viewmodels/report"></compose>
        </article>
        <article class="column">
          <compose view-model="./viewmodels/candidates"></compose>
        </article>
        <article class="column">
          <compose view-model="./viewmodels/stats"></compose>
        </article>
      </div>
    </section>
  </div>
</template>
```

```
export class App {
  email = 'marge@simpson.com';
  password = 'secret';

  loggedIn = false;

  login() {
    console.log(`Logging in ${this.email}`);
    this.loggedIn = true;
  }
}
```

- show.bind selectively reveals a view or view/view-model pair
- loggedIn -> false: show login view
- loggedIn -> true: show remaining views