Outline

- **How backbone.js uses the server**
- **Same origin policy**
  - Cross-origin resource sharing (CORS)
- **Collection requests**
- **Model requests**
- **The server**
- **Backbone.sync**
- **Backbone.localStorage**

How Backbone Uses the Server

- **Backbone uses RESTful web requests to synchronize data to and from a server**
- **Backbone's data server does not have to be the server that served the page**
  - But the same origin policy applies

Same Origin Policy

The same origin policy prevents scripts from accessing resources belonging to another site

Same Origin Policy

- Origin is application layer protocol + domain name + port number

`http://localhost:3000`

`http://localhost:3000/a`	==	`http://localhost:3000/b`
`http://localhost:3001/a`	!=	`http://localhost:3000/a`
`http://localhost:3000/a`	!=	`https://localhost:3000/a`

Cross-origin Resource Sharing (CORS)

- Technology that allows cross-origin requests
- Uses special http headers to specify the set of valid origins
- Alternative to jsonp
- Supported by most modern browsers

The Server

- Backbone defines a HTTP persistence protocol, but does not include a server
- Use any http server that can implement Backbone's restful protocol

backbone-server

- Simple, transient backbone server
- https://github.com/liammclennan/backbone-server
- Or write a specialized api with asp.net mvc web api, ruby on rails, node.js etc

Collection Requests (cont.)

- **people.create({ name: “Tom”, age: 50 });**
  - Saves to the server and adds to the collection

Request

```
POST /people HTTP/1.1
Host: localhost:3002

{"name":"Tom","age":50}
```

Response

```
HTTP/1.1 200 OK

{"id":3}
```

Collection Requests

- **people.fetch()**
  - Fetch the collection from the server

Request

```
GET /people HTTP/1.1
Host: localhost:3002
```

Response

```
HTTP/1.1 200 OK

[{"name":"Sarah","age":64,"id":0},
 {"name":"John","age":19,"id":1}]
```

Model Requests

- **person.fetch()**
  - Reset the model's state from the server

Request

```
GET /people/1 HTTP/1.1
Host: localhost:3002
```

Response

```
HTTP/1.1 200 OK

{"name":"Sarah","age":64,"id":1}
```

Model Requests (cont.)

- **person.save()**
  - Create or update depending upon person.isNew()
  - Create is the same as collection.create()

Request (update)

```
PUT /people/1 HTTP/1.1
Host: localhost:3002

{"name":"Tom","age":50}
```

Response (update)

```
HTTP/1.1 200 OK
```

Model Requests (cont.)

- **person.destroy()**
  - Deletes the model on the server and removes it from its client-side collection

Request

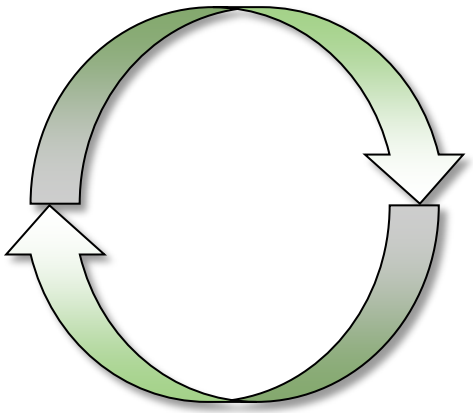
```
DELETE /people/1 HTTP/1.1
Host: localhost:3002
```

Response

```
HTTP/1.1 200 OK
```

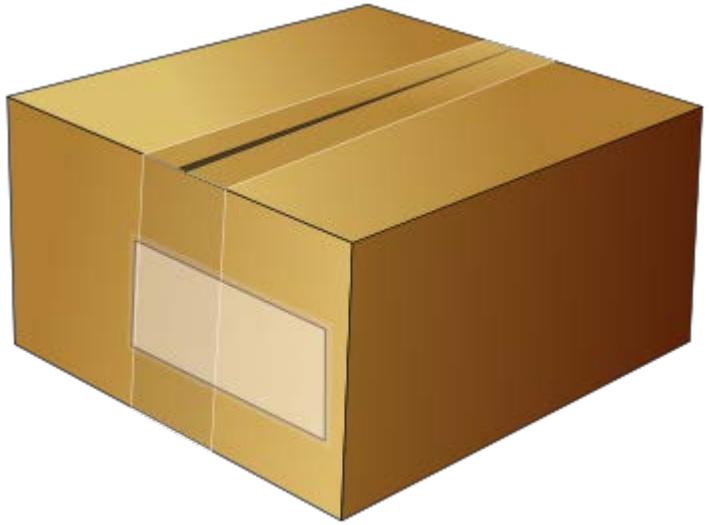
Backbone.sync

- **A function that interfaces between backbone and the server**
- **Implements create, read, update and delete behavior**
  - Can be overridden globally, per collection, or per model



Backbone.localStorage

- A backbone ‘plugin’ that uses local storage for persistence
- replaces Backbone.sync with a local storage implementation



Summary

- Backbone.js synchronizes with the server via a simple restful protocol.
- Backbone's synchronization behavior is implemented by the Backbone.sync function
- The 'same origin policy' prevents a backbone.js application from making requests to different origins
- Backbone.js' default syncing behavior maps create, read, update and delete operations to asynchronous restful server requests.