TDD on the Server

Nate Taylor
http://taylonr.com
@taylonr

hardcore dev and IT training

Why Start With Server?

Common Place to Test

Mostly Logic

Easier to Automate

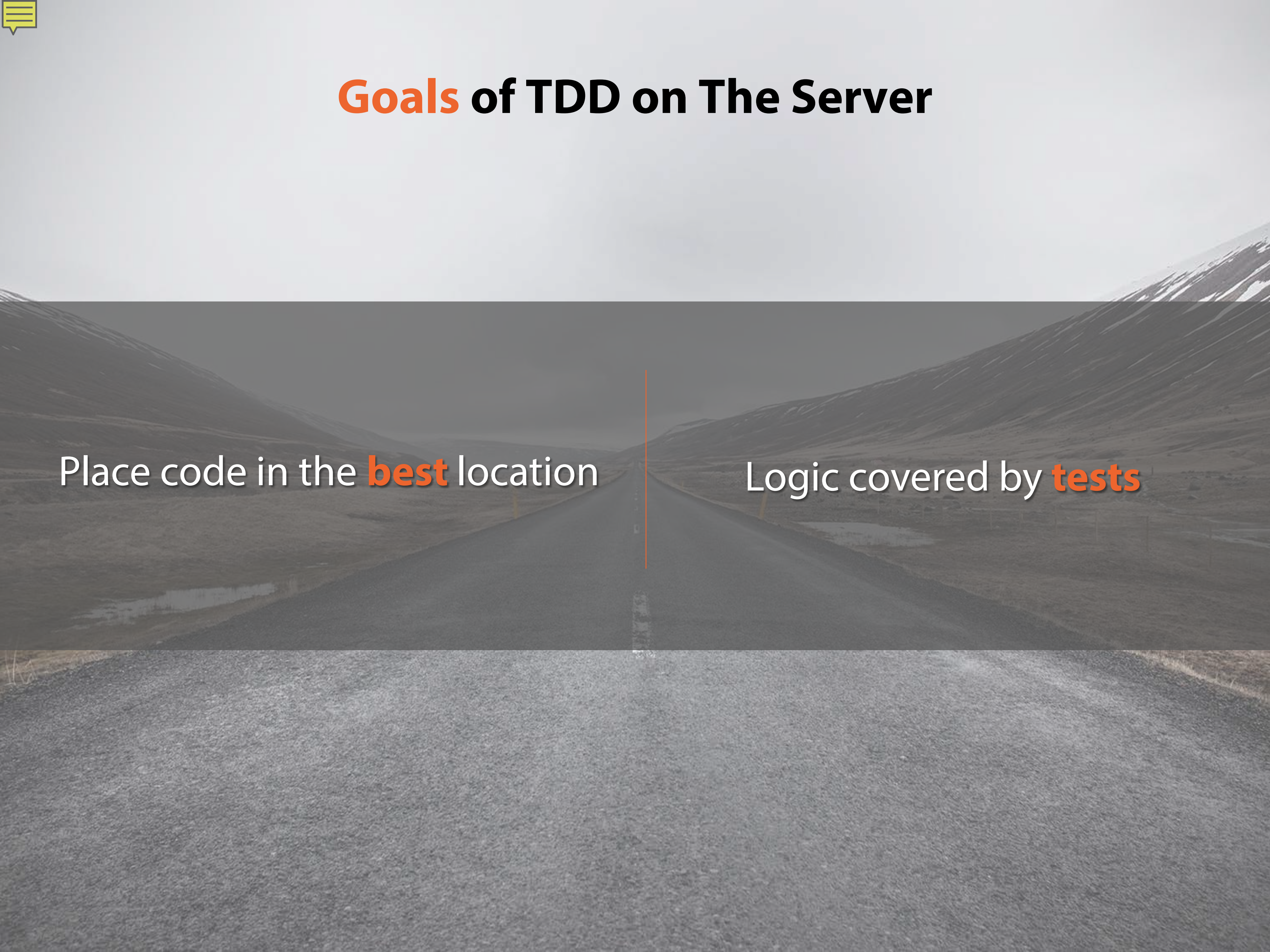
Challenges On the Server

Routing

Database Interactions

Knowing What To Test

Goals of TDD on The Server



Place code in the **best** location

Logic covered by **tests**

What Are We Building?

Rating Application

- Allow people to rate an event
- Provide comments about their rating
- See how others rate events

Starting With the Model

- **Name**
  - Railroad Days
  - Settler’s Days
- **Description**
  - Historical railroad landmarks.
  - Celebration of our town’s founding. Complete with food & rides
- **Average Score**
- **Ratings**
- **Comments**

Pseudo-Model

```
{
   name: “Railroad Days”,
   description: “Historical railroad
   landmarks”,
   averageRating: 3.7
   ratings: [
     rating: 4,
     comment: “Ride the steam engine!”
   ]
}
```

TO THE CODE!

More Than Models

Interacting with Models

Models by themselves not helpful

Client needs access

Need API routes

Structure of API

- **Routes**
  - /events – Access information about events
  - /events/{id}/reviews– Access reviews for an event
- **Actions**
  - GET – Fetch data
  - POST – Add data
  - PUT – Update data
  - DELETE – Remove data

Why not API **First**?

Keep focus on what is unique to **this** application

Keep responsibilities **separate**

Next logical building block

Sample API Request

```
Request Url: api.example.com/events

Request Method: GET

Returns: [

   {eventModel},

   {eventModel}

   …

]
```

```
TO THE CODE!
```

GET Events

```
Request Url: api.example.com/events

Request Method: GET

Returns: [

   {eventModel},

   {eventModel}

   …

]
```

GET Single Event

```
Request Url: api.example.com/events/{id}

Request Method: GET

Returns: {

   id: {id},

   name: {eventName},

   ratings: [{rating}, {rating}]

}
```

Final Step of API

Routes

- How consumer gets to data

Controller

- Go-between consumer & model

Models

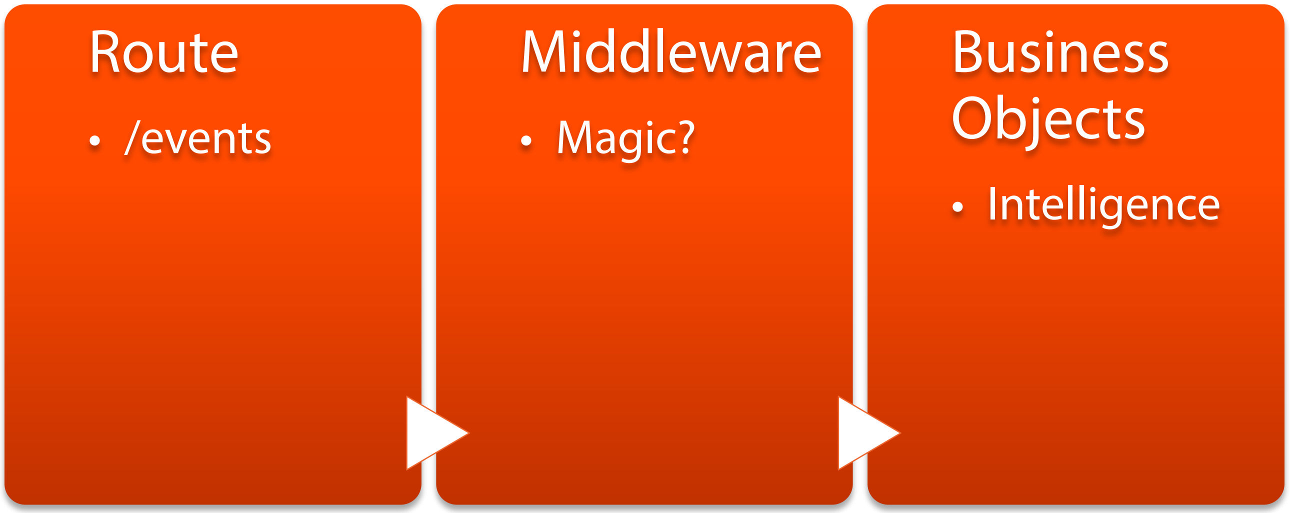
- Logic
- Business Objects

Why Test Routes?

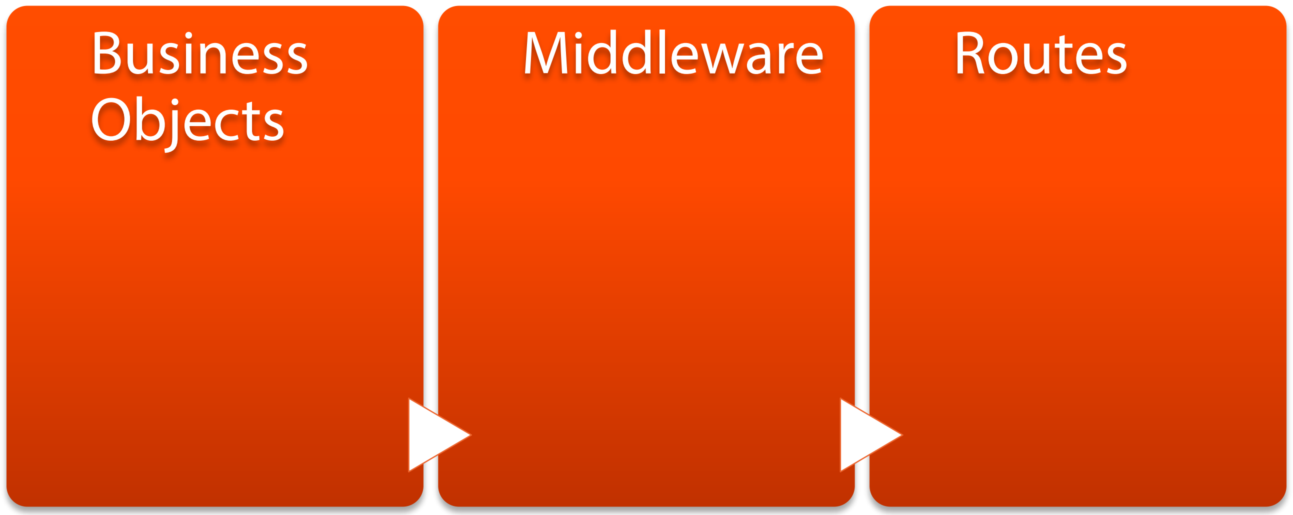
- Not required
- Key Aspect of Application
- Can Get Tricky

What Have We Learned?

How Consumer’s See API



How We TDD an API



Test Fake

An object or method that allows you to simulate how other code should perform.

Summary

TDD On Server Straight Forward

Fake/Mock as Much as Possible

Don’t Forget to Refactor