User Params

Massaging user data

Level 3 - Part II

Routes don’t match all cases

Current implementation only matches on **exact** Block name

```
$ curl -i http://localhost:3000/blocks/Fixed

HTTP/1.1 200 OK
"Fastened securely in position"
```

```
$ curl -i http://localhost:3000/blocks/fixed

HTTP/1.1 404 Not Found
"No description found for fixed"
```

does not match
on lower case

Normalizing the request parameter

Let’s split the steps to improve code clarity

app.js

```
...
var blocks = {
  'Fixed': 'Fastened securely in position',
  'Movable': 'Capable of being moved',
  'Rotating': 'Moving in a circle around its center'
};
app.get('/blocks/:name', function(request, response) {
 var description = blocks[request.params.name];
  ...



});
...
```

doing two things at once

Normalizing the request parameter

When one line does only **one thing**, it makes code easier to understand

app.js

```
...
var blocks = {
  'Fixed': 'Fastened securely in position',
  'Movable': 'Capable of being moved',
  'Rotating': 'Moving in a circle around its center'
};
app.get('/blocks/:name', function(request, response) {
 var name = request.params.name;
 var block = name[0].toUpperCase() + name.slice(1).toLowerCase();

  ...

});
...
```

first character to upper case and remaining characters to lowercase

Normalizing the request parameter

Use the **normalized** block name to look up its description

app.js

```
...
var blocks = {
  'Fixed': 'Fastened securely in position',
  'Movable': 'Capable of being moved',
  'Rotating': 'Moving in a circle around its center'
};
app.get('/blocks/:name', function(request, response) {
 var name = request.params.name;
 var block = name[0].toUpperCase() + name.slice(1).toLowerCase();
 var description = blocks[block];
 if (!description) {
   ...
});
...
```

block **name** is now in the same format as the properties in the **blocks** object

Supporting any url argument case

```
$ curl -i http://localhost:3000/blocks/Fixed

HTTP/1.1 200 OK
"Fastened securely in position"
```

```
$ curl -i http://localhost:3000/blocks/fixed

HTTP/1.1 200 OK
"Fastened securely in position"
```

```
$ curl -i http://localhost:3000/blocks/fiXeD

HTTP/1.1 200 OK
"Fastened securely in position"
```

any case is now properly supported

Same parameter used on multiple routes

app.js

```
var blocks = { ... };

var locations = {
  'Fixed': 'First floor', 'Movable': 'Second floor', 'Rotating': 'Penthouse'
};

app.get('/blocks/:name', function(request, response) {
  var name = request.params.name;
  var block = name[0].toUpperCase() + name.slice(1).toLowerCase();
  ...
});

app.get('/locations/:name', function(request, response) {
  var name = request.params.name;
  var block = name[0].toUpperCase() + name.slice(1).toLowerCase();
  ...
});
```

duplication

Extracting duplication to app.param

The **app.param** function maps placeholders to callback functions.
It's useful for running **pre-conditions** on dynamic routes.

app.js

```
var blocks = { ... };

var locations = {
  'Fixed': 'First floor', 'Movable': 'Second floor', 'Rotating': 'Penthouse'
};

app.param('name', function(request, response, next) {
  var name = request.params.name;
  var block = name[0].toUpperCase() + name.slice(1).toLowerCase();


});
...
```

called for routes which include the **:name** placeholder

Setting properties on the request object

Properties set on the **request** object can be accessed from all subsequent routes in the application

app.js

```
var blocks = { ... };

var locations = {
  'Fixed': 'First floor', 'Movable': 'Second floor', 'Rotating': 'Penthouse'
};

app.param('name', function(request, response, next) {
  var name = request.params.name;
  var block = name[0].toUpperCase() + name.slice(1).toLowerCase();

  request.blockName = block;

  next();
});
...
```

can be accessed from other routes in the application

must be called to resume request

Accessing custom properties on request

We can read properties on **request** which were set on **app.param**

app.js

```
...
app.param('name', function(request, response, next) {
  ...
});

app.get('/blocks/:name', function(request, response) {
  var description = blocks[request.blockName];
  ...
});

app.get('/locations/:name', function(request, response) {
  var location = locations[request.blockName];
  ...
});
```

Dynamic routes with curl

Refactoring improved our code without affecting the output

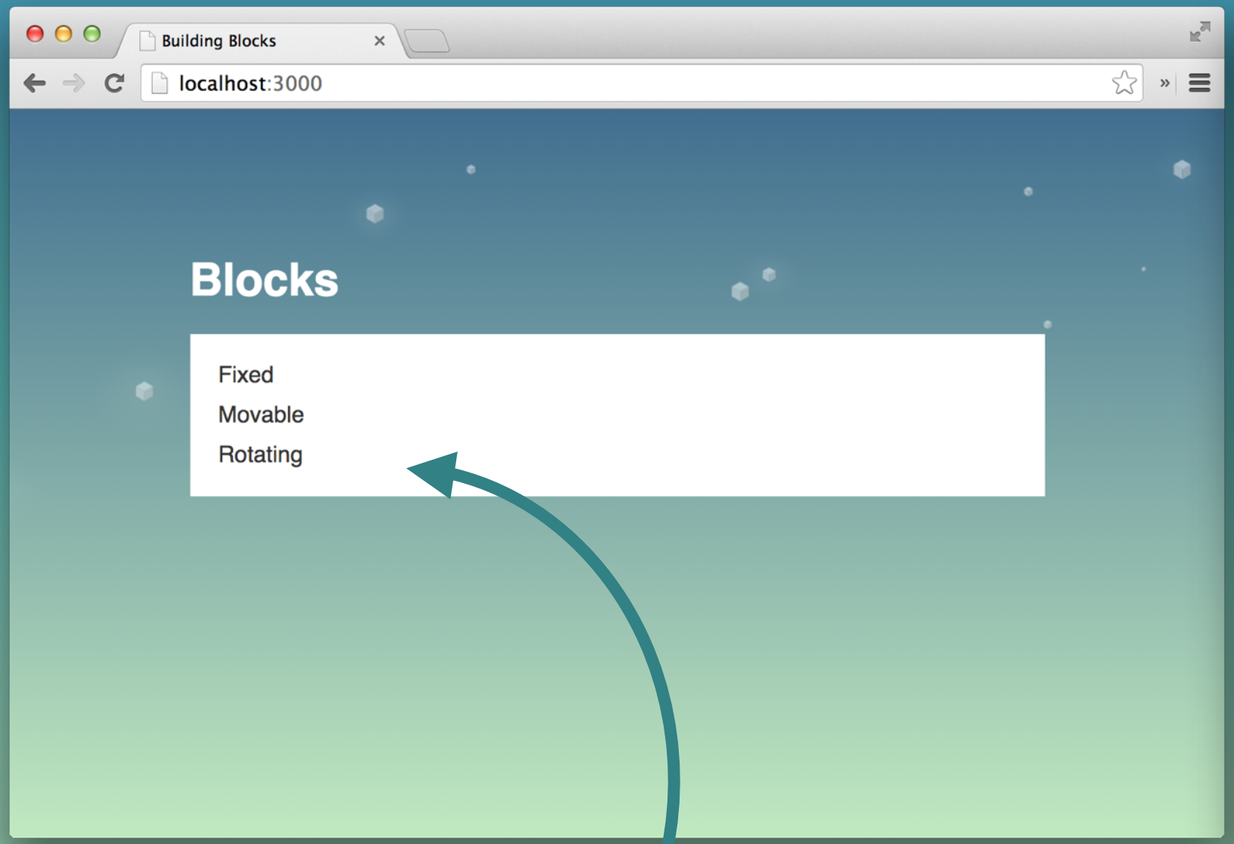
```
$ curl -i http://localhost:3000/blocks/fixed

HTTP/1.1 200 OK
"Fastened securely in position"
```

same result
as before

```
$ curl -i http://localhost:3000/locations/fixED

HTTP/1.1 200 OK
"First floor"
```

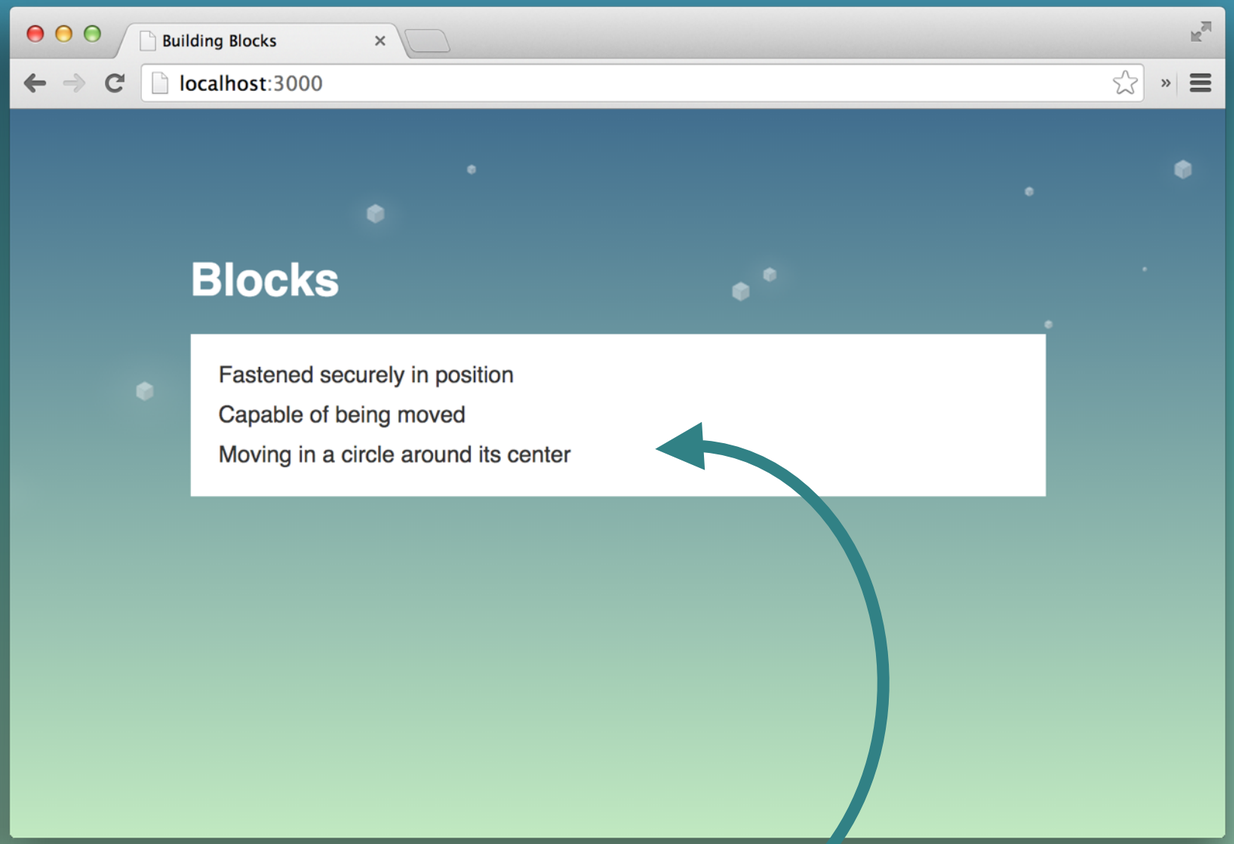
Something looks different

Initially:



listing block **names**

Now:



listing **descriptions**
instead of names

Breaking the initial listing of Blocks names

Initially:

app.js

```
var blocks = ['Fixed', 'Movable', 'Rotating'];
```

Now:

moved from **Array** to **object**

app.js

```
var blocks = {
  'Fixed': 'Fastened securely in position',
  'Movable': 'Capable of being moved',
  'Rotating': 'Moving in a circle around its center'
};
```

Fixing Block names

Responding with **object** instead of **Array** is what broke our route

app.js

```
var blocks = {
  'Fixed': 'Fastened securely in position',
  'Movable': 'Capable of being moved',
  'Rotating': 'Moving in a circle around its center'
};

app.get('/blocks', function(request, response) {
  response.json(blocks);
});

...
```

serializes blocks **object**

Fixing Block names

The **Object.keys** function returns an Array with the object's **properties**

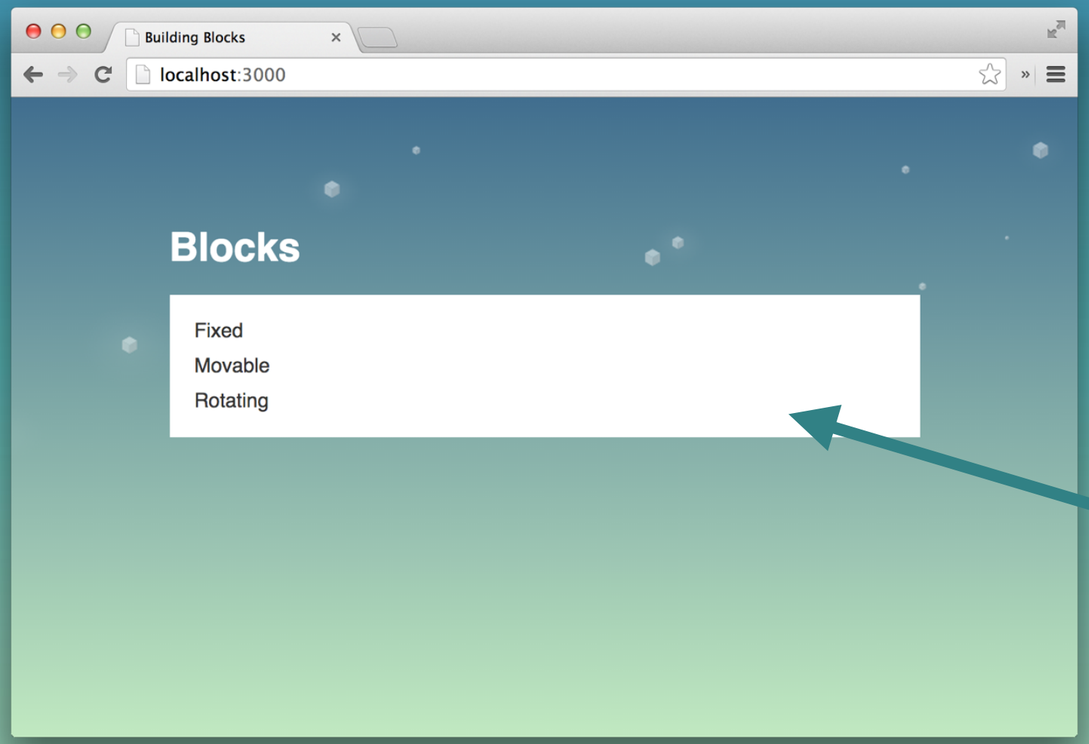
app.js

```
var blocks = {
  'Fixed': 'Fastened securely in position',
  'Movable': 'Capable of being moved',
  'Rotating': 'Moving in a circle around its center'
};

app.get('/blocks', function(request, response) {
  response.json(Object.keys(blocks));
});

...
```

returns **properties** from the blocks object

Responding with Block names

Now:




```
$ curl -i http://localhost:3000/blocks

HTTP/1.1 200 OK
["Fixed","Movable","Rotating"]
```

back to listing
Block **names**