Outline

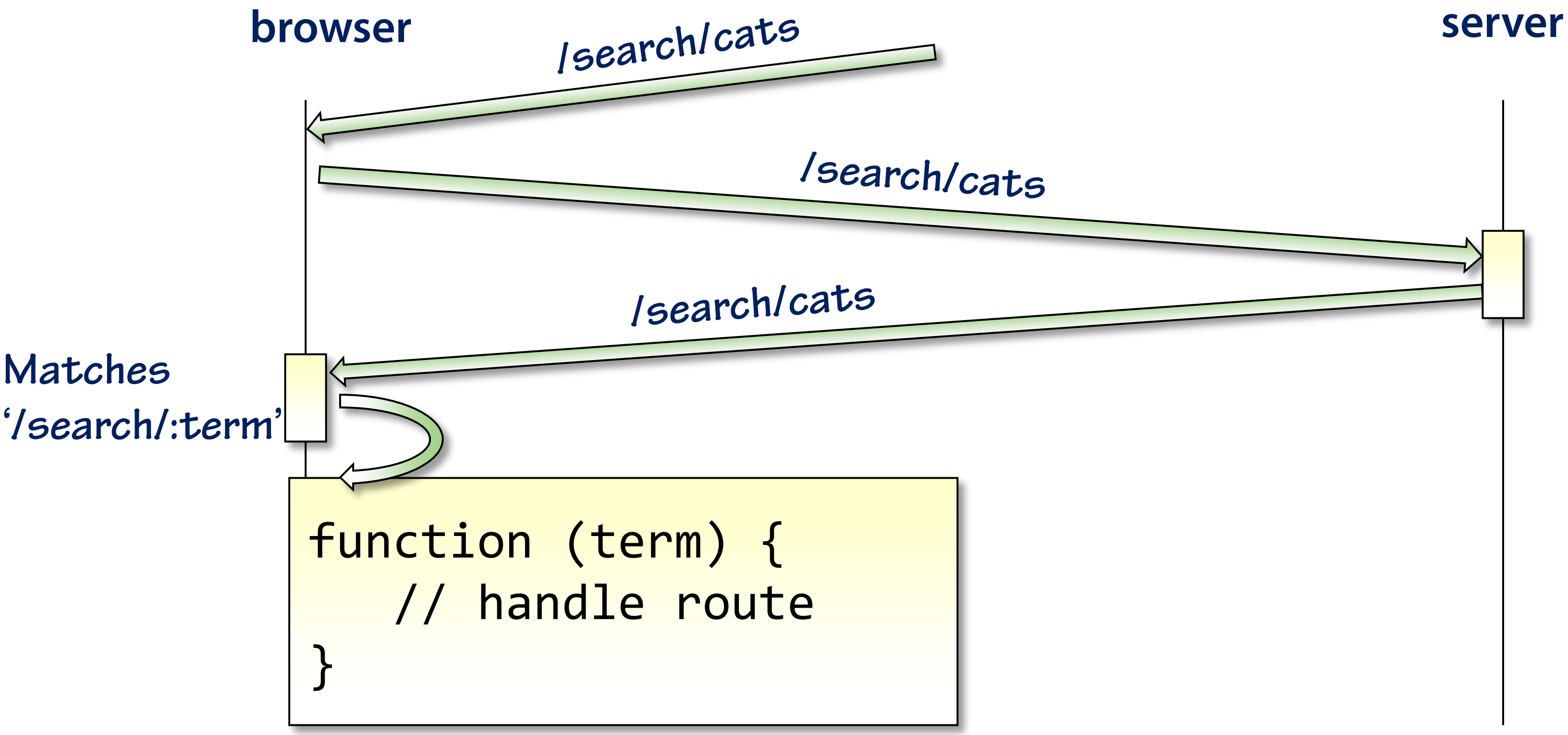
- Client-side routing
- How to define routes
- Triggering routes (navigate)
- Html5 history pushstate
- Hash fragments
- Search Engine Indexability

Routing

- Client-side routes are a way to trigger a function when the browser url changes
- Backbone routing includes parsing of the url and matching the url to the correct route handler
- Don’t use routes like MVC actions
- Each route results in two different scenarios

Client-side Routing

- Routing a browser initiated request



browser

server

/search/cats

/search/cats

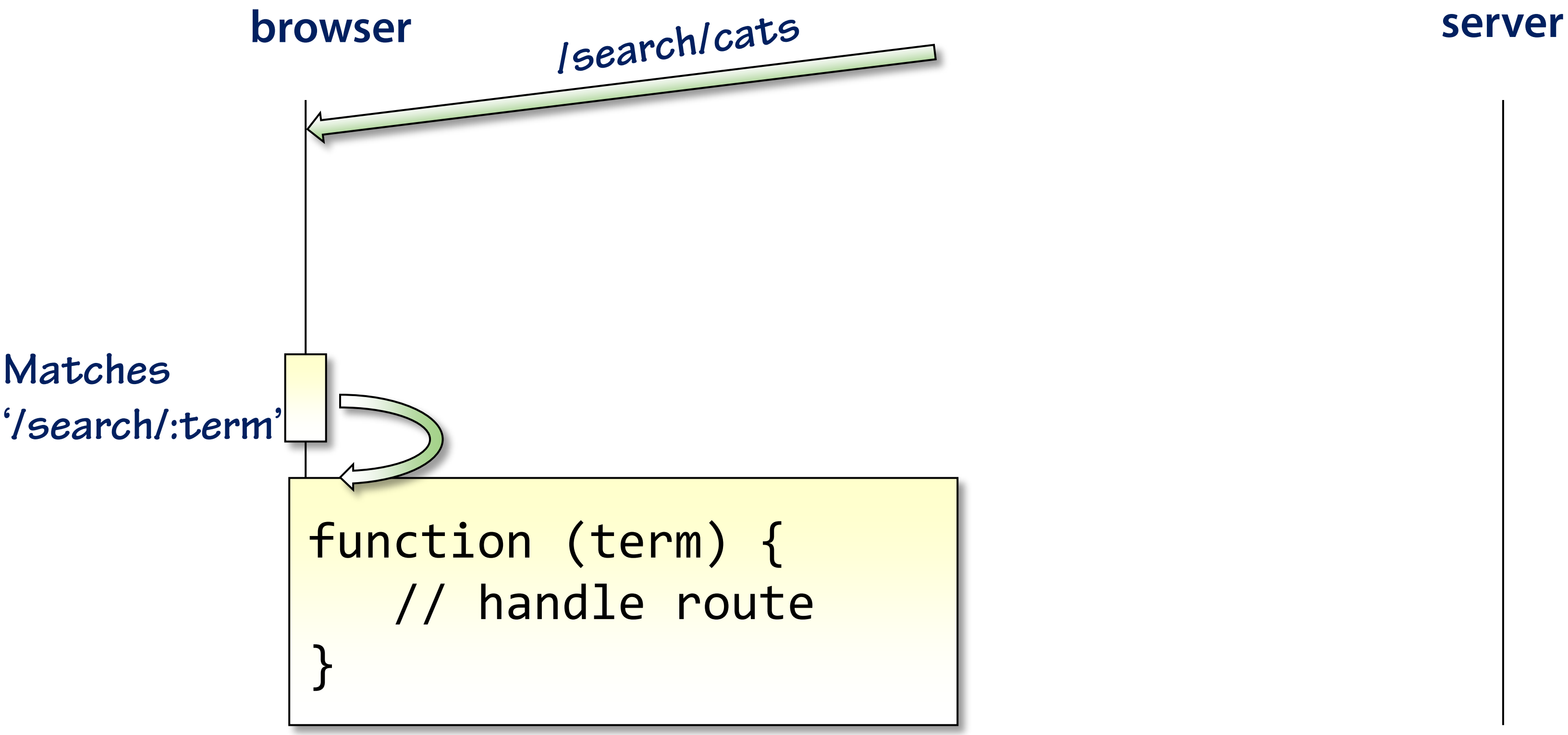
/search/cats

Matches '/search/:term'

```
function (term) {
   // handle route
}
```

Client-side Routing

- Routing a client request



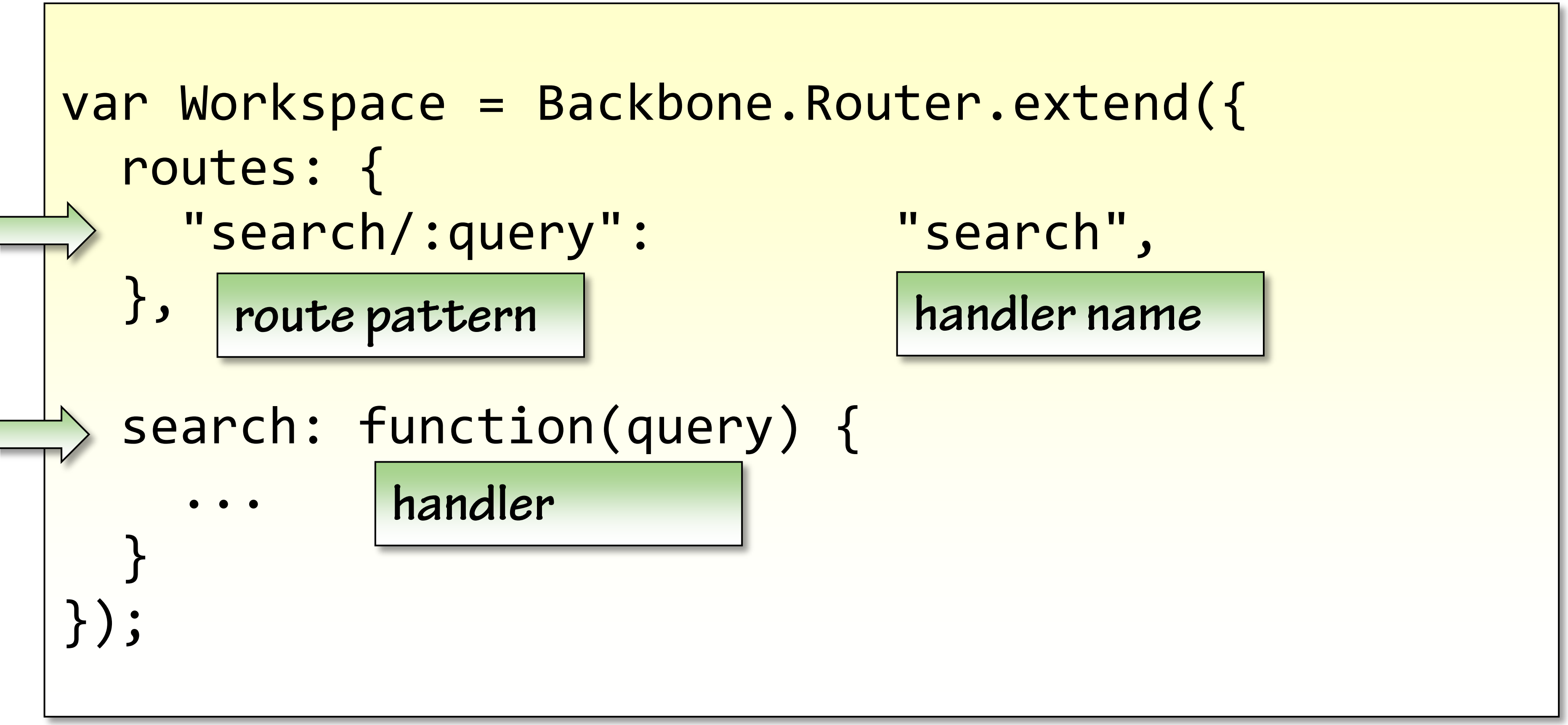
```
function (term) {
   // handle route
}
```

Document Router

```
Given a set of documents
When a user selects a document
Then that document is displayed
```

Defining routes

- **Define routes by defining a type that extends Backbone.router**



```
var Workspace = Backbone.Router.extend({
  routes: {
    "search/:query":          "search",
  },

  search: function(query) {
    ...
  }
});
```

route pattern

handler name

handler

navigate

- navigate is the backbone function for updating the browser’s address and triggering routing

```
var router = new Workspace();
Backbone.history.start();

router.navigate('search/cats', {
    trigger: true
});
```

- The first parameter is the new url path
- The second parameter tells Backbone if it should trigger routing

Push state

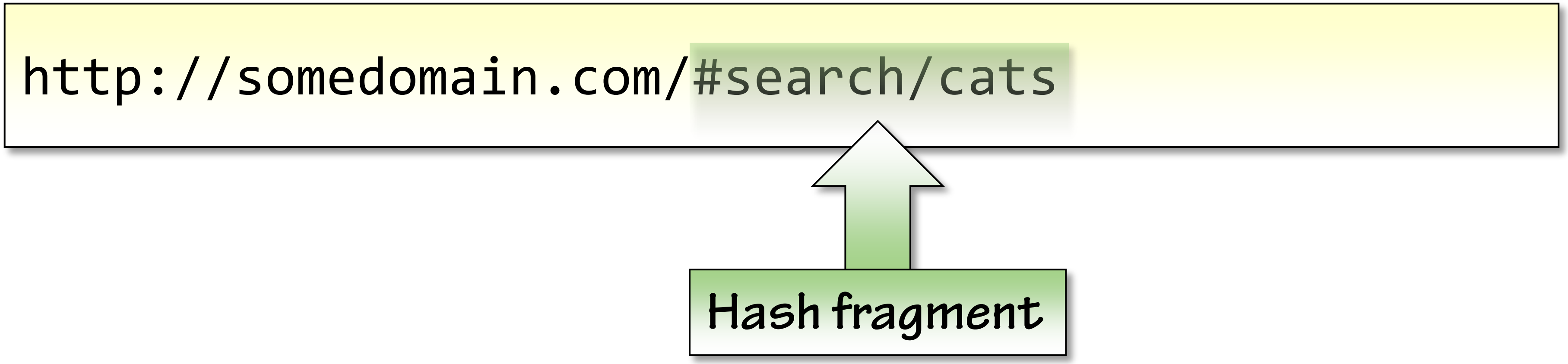
- **The Html5 history api introduced a way to change the browser url without reloading the page**

```
window.history.pushState(…);
```

- **Browser support is patchy**
  - Currently not supported by any version of Internet Explorer
  - caniuse.com

Hash Fragments

- If the browser does not support the Html5 history api Backbone will use hash fragments
- Browsers have always allowed javascript to modify the page url by appending a hash followed by a string

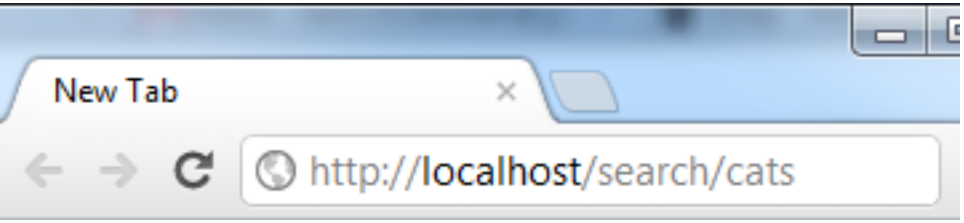


```
http://somedomain.com/#search/cats
```

Hash Fragments (cont.)

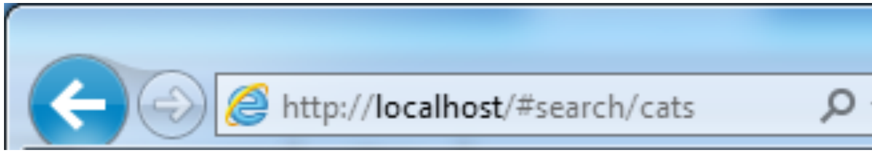
```
router.navigate('search/cats', { trigger: true });
```

Html5 history api



http://localhost/search/cats

Hash fragments



http://localhost/#search/cats

- Both will trigger the ‘search/:query’ route

Search Engine Indexability

- Markup that is rendered on the client will not be indexed by search engines.
- Option 1: render content on the server
- Option 2: #! Urls

```
Backbone.history.start({ pushState: false });
router.navigate('!search/cats', { trigger: true });

// generated url -> /#!search/cats
// google converts to ->
       /_escaped_fragment_=search/cats
```

https://developers.google.com/webmasters/ajax-crawling/docs/specification

Summary

- Don’t use routing like mvc controllers
- Defining route – patterns and handlers
- Html 5 history and pushstate
- Hash fragments