Remoting in iOS

An introduction to calling remote services on mobile Apple devices

Overview

Introduction: what is remoting?

A first attempt...

Pitfalls?

An improved approach

Pitfalls, again?

The correct approach

Q&A

Introduction

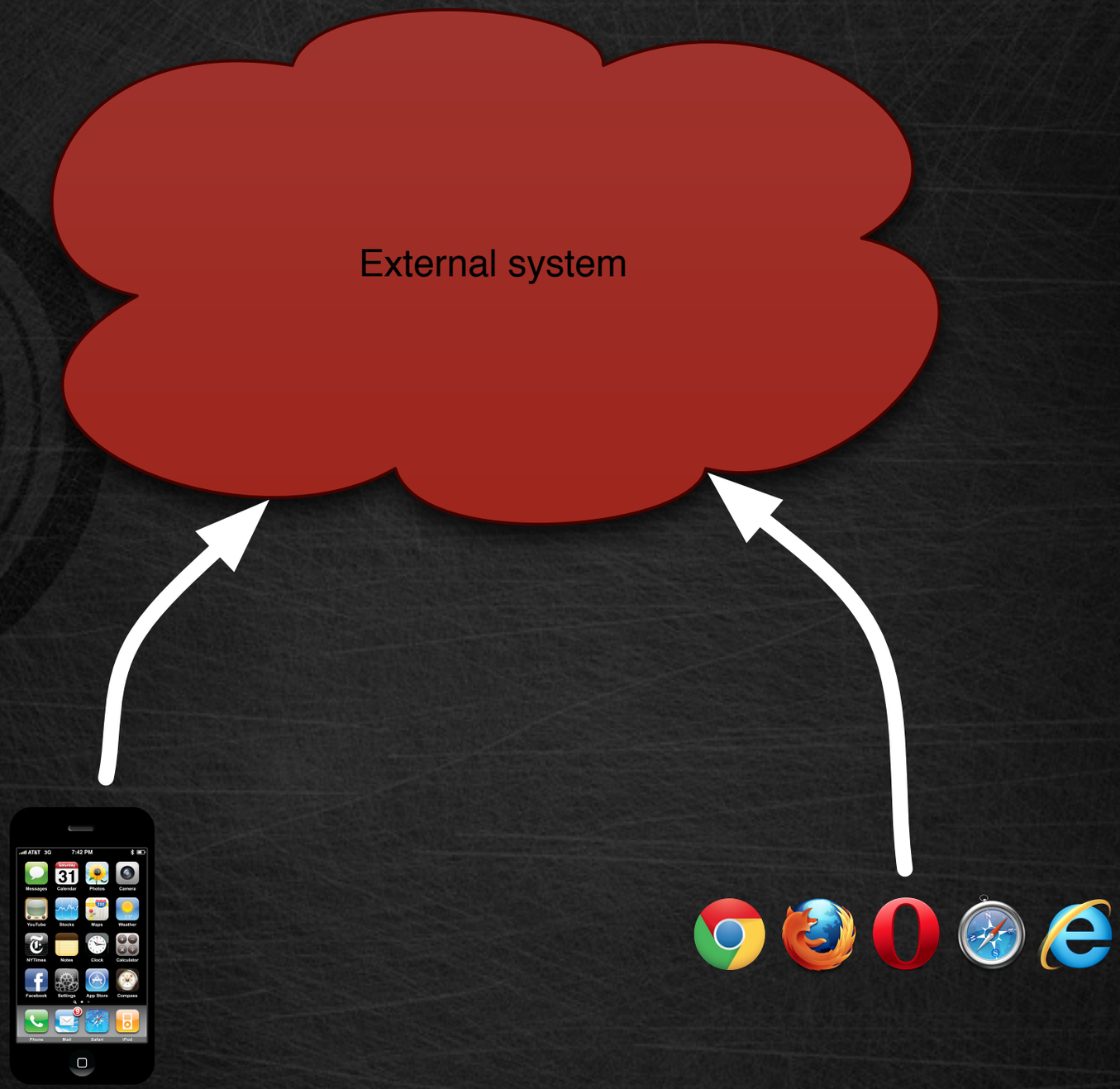
Remoting

What is remoting?

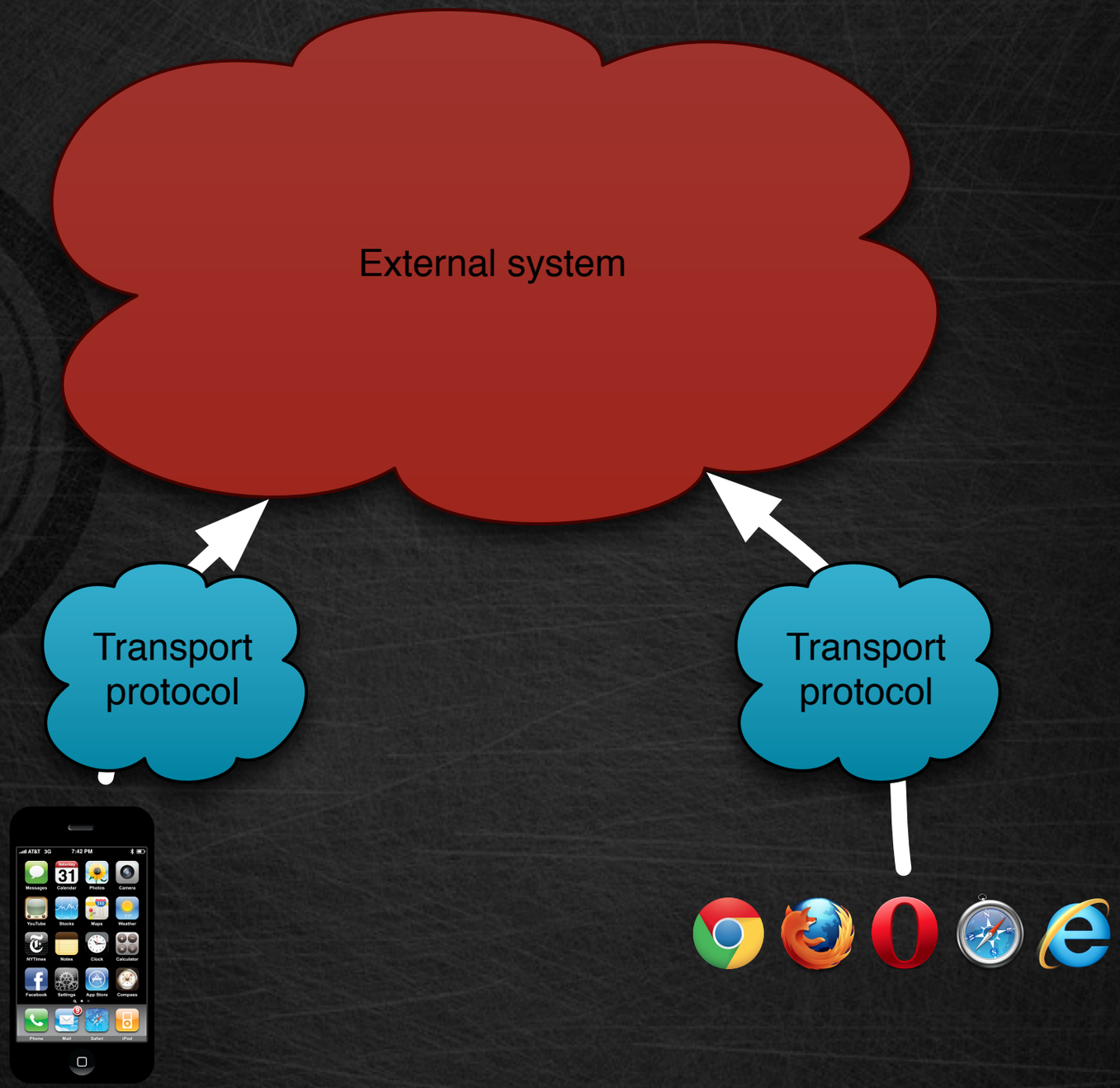
What are the building blocks?

In which format is data represented?

What is remoting?



Data exchange



REST is our friend

Big Bang App data in the cloud

http://idamf-restdemo.herokuapp.com/

List of persons: /app/persons.json

Detail of a person: /app/persons/{id}.json

GET, PUT, POST, DELETE

A first attempt...

Pop quiz

Which classes can we use, out-of-the-box, in iOS to communicate over HTTP?

NSURLConnection, NSURLRequest & HTTP extensions

NSURLConnection and company

Loads URL requests

Both synchronous and asynchronous

Asynchronous calls are handled with a delegate

See reference guide!

Voorbeeld

```
NSURL *url = [NSURL URLWithString:@"http://..."];
NSURLRequest *request = [NSURLRequest requestWithURL:url];

NSURLResponse *response;
NSError *error;
NSData *data = [NSURLConnection sendSynchronousRequest:request
                                     returningResponse:&response
                                                 error:&error];
```

Exercise

Add a method to MasterViewController to load person data using an URL Connection

URL = http://idamf-restdemo.herokuapp.com/app/persons

Log the result with NSLog (use [NSString stringWithUTF8String:data.bytes])

Pop quiz

What is the easiest way to send/receive data in iOS?

JSON data, using NSJSONSerialization (since iOS 5.0, otherwise: use JSONKit)

Example

```
id jsonObject = [NSJSONSerialization JSONObjectWithData:data
                                                options:...
                                                  error:&error];
```

Exercise

In the already added method, add more code to transform NSData to a JSON object.

Then transform this JSON object so that it fits in the _persons array!

Add a refresh button to the toolbar, and make sure, when tapped, it reloads the data again

Summary

In iOS, it is fairly easy to load data from an external system over HTTP

JSON or XML can be transformed into objects using default iOS classes

Pitfalls?

Problem?

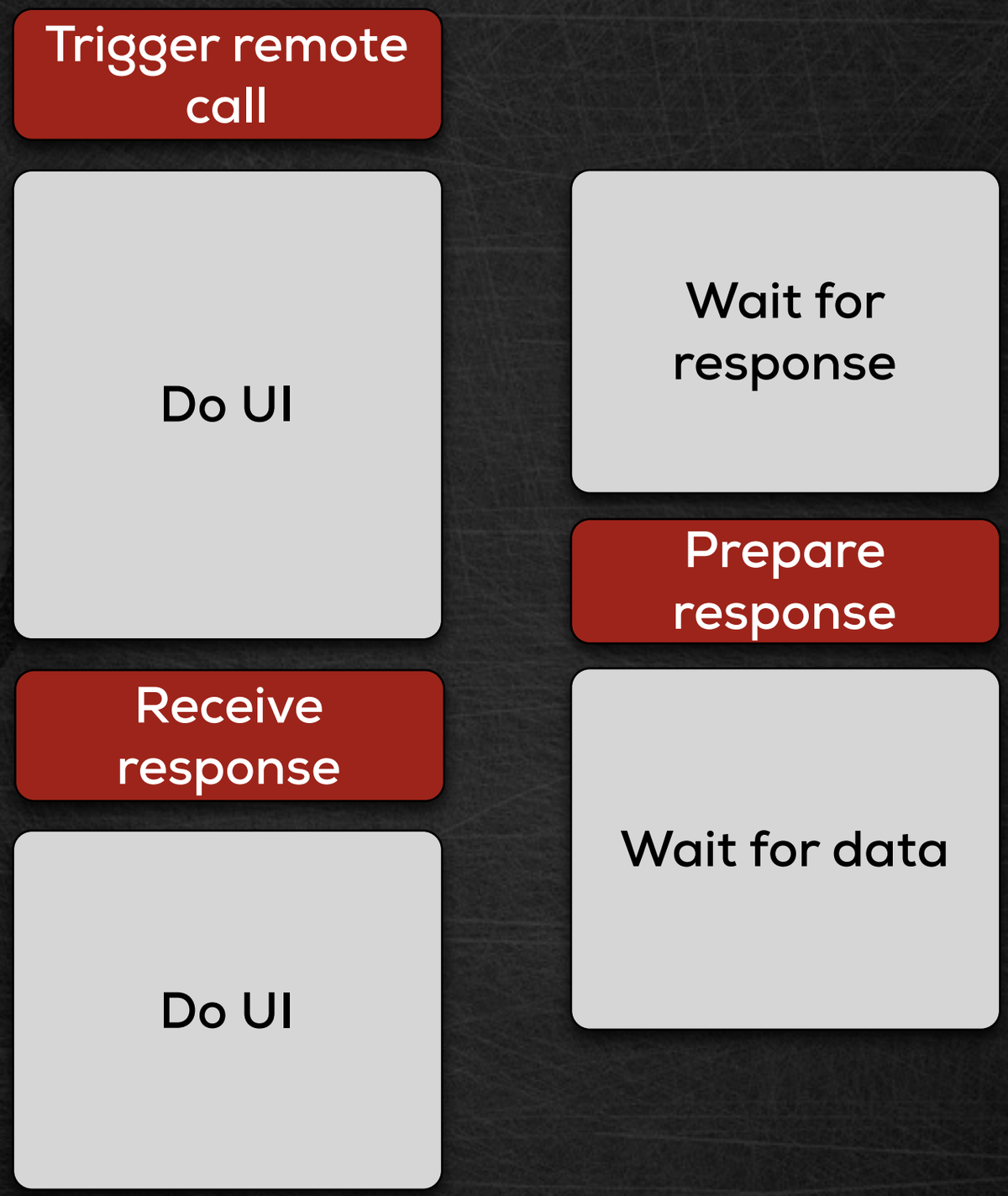
We are blocking the GUI!

Try it: add “?timeOut=5000” to the URL we are requesting. Also try to reload and then navigate to the detail view. Does this work?

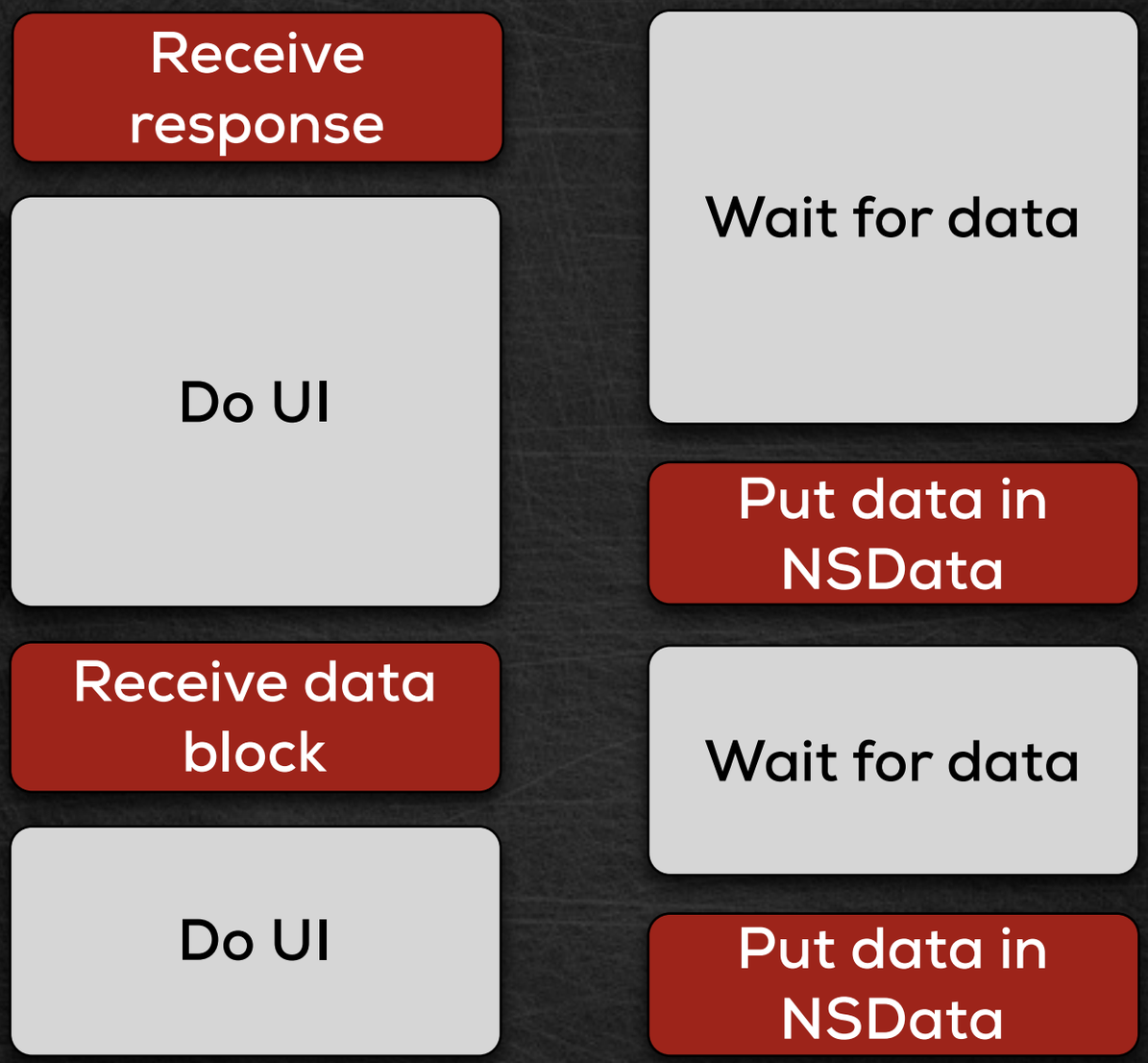
An improved approach

Any ideas / thoughts?

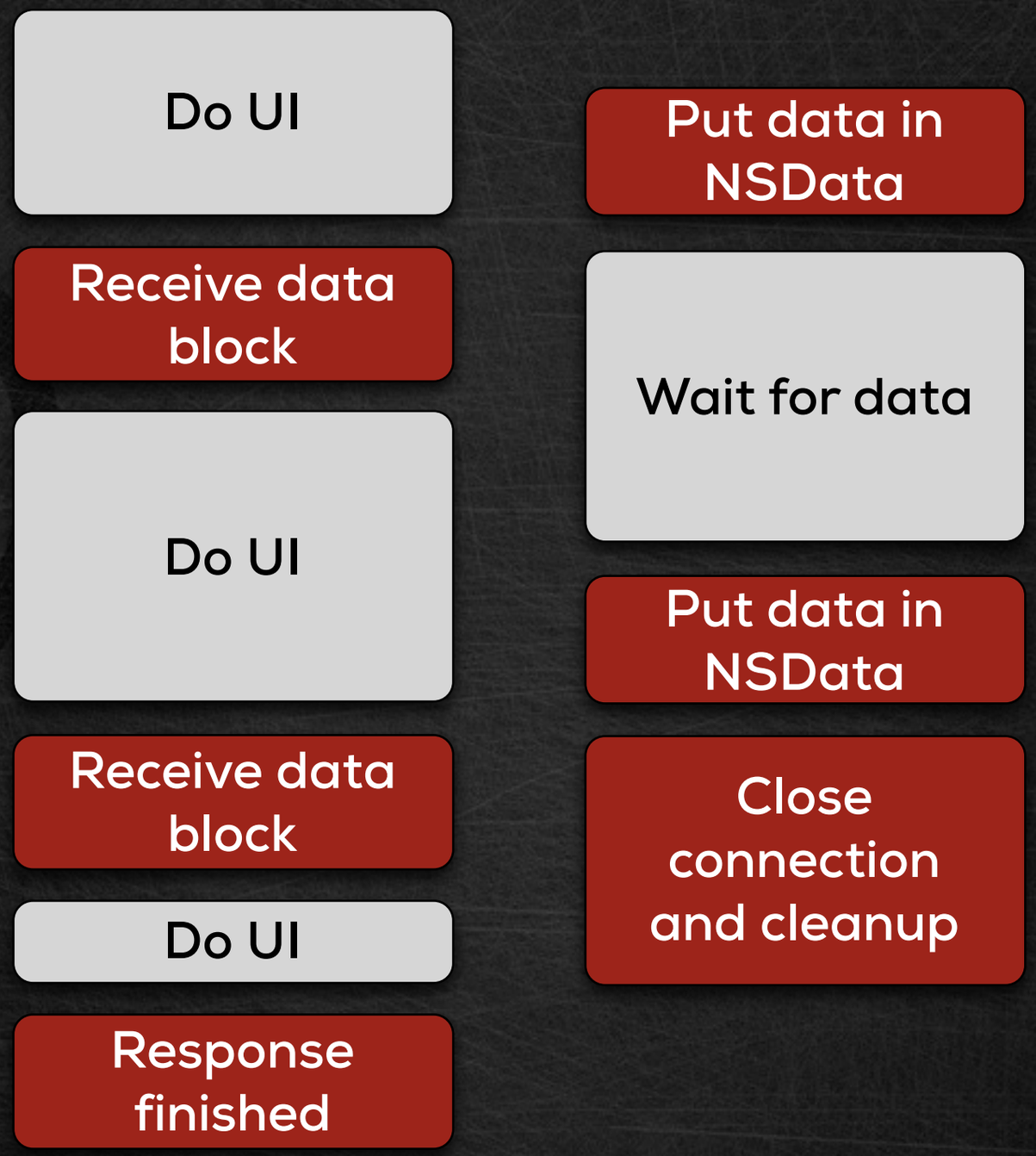
Asynchronous handling



Asynchronous handling



Asynchronous handling



NSURLConnection(Data) Delegate

Contains methods for

- Dealing with an initial answer
- Handling data parts
- Handling the end of a response
- Handling errors

See reference guide!

Example

```
//Header
@interface MasterViewController :
    UITableViewController<NSURLConnectionDelegate,
                          NSURLConnectionDataDelegate>
    ...
@end

//Call
{
   ...
   [NSURLConnection connectionWithRequest:request
                                 delegate:delegate];
   ...
}
```

Example

```
//Delegate
- (void)connection:(NSURLConnection *) connection
didReceiveResponse:(NSURLResponse *) response {
    //Look at the response, is everything ok?
}

- (void)connection:(NSURLConnection *) connection
  didReceiveData:(NSData *) data {
    //Add data to data block
}

- (void)connectionDidFinishLoading:(NSURLConnection *)
connection {
    //Handle the result
}

- (void)connection:(NSURLConnection *) connection
  didFailWithError:(NSError *) error {
    //Report an error
}
```

Dealing with initial answer

```
- (void)connection:(NSURLConnection *)connection
didReceiveResponse:(NSURLResponse *)response {
  //Look at the response, is everything OK?
  if ([response isKindOfClass:NSHTTPURLResponse.class]) {
    NSHTTPURLResponse *httpResponse = (NSHTTPURLResponse *)response;
    if (httpResponse.statusCode != 200) {
      //Report an error!
    } else {
      //Create a data block, so we can use it later on
      _responseBody = [[NSMutableData alloc] init];
    }
  }
}
```

Handling data parts

```
- (void)connection:(NSURLConnection *)connection
    didReceiveData:(NSData *)data {
        //Add data to data block
        [_responseBody appendData:data];
}
```

Wrapping it up

```
- (void)connectionDidFinishLoading:(NSURLConnection *) connection
{
    //Handle result, transform it to something useful
    if (_responseBody) {
        ...
    }
}
```

Exercise

Refactor to use the asynchronous approach

Check by adding the timeOut parameter once again. Does it block the ui? (can you for instance navigate to the detail?)

Summary

Use asynchronous calls when doing long running operations, this prevents blocking the UI thread.

When doing asynchronous calls in iOS, we use delegates

All done!

Well... Not yet!

Pitfalls, again!

At first sight, everything seems to be OK, but what happens when we fire multiple requests at same time?

The correct approach

Any ideas / thoughts?

Thread-safety!

If a connection delegate instance is reused by more connections, we get unwanted behavior

Data from 2 different requests is put into 1 data container

Using “self” as delegate is most of the time a bad idea

So how should we do it?

Solution

Create a new class

Create an instance of this class, every time a new request is made

Make sure we can return results and errors to the controller

Sending back data?

A lot of possibilities here!

We will do the most simple and direct solution:

Add a protocol and adopt it on the ViewController

Make our new connection delegate class work with the new protocol

Example

```
@protocol PersonDataKeeper <NSObject>

@property (nonatomic, strong) NSArray *persons;

- (void) handleError:(NSError *) error;

@end
```

```
@interface PersonConnectionDelegate :
NSObject<NSURLConnectionDelegate, NSURLConnectionDataDelegate>

@property (nonatomic, weak) id<PersonDataKeeper> dataKeeper;

@end
```

Exercise

Create the data keeper protocol

Create the new connection delegate

Change the MasterViewController so that it becomes a data keeper, instead of a connection delegate

Change the code so that it uses our new connection delegate and our data keeper methods

Improving data traffic

Compress!

Avoid processing the same data over and over again!!

Exercise

Change the connection delegate so that it checks for the ETag header and make sure the "data keeper" is notified of the ETag

Also, add a check for the 304 response status code, and make sure the "data keeper" is notified when this occurs

Q & A