SYDNEY ETHEREUM DEV.WORKSHOP

METEOR & WEB3

EMMA POPOSKA

WE'LL TALK ABOUT

- Installing web3
- asynchronous callbacks
- get transaction data
- invoke a transaction
- listening for events
- Integration with Meteor

WHY METEOR+WEB3

One of the most used stack that DApp developers use is the Meteor+Web3. Meteor will help you to quickly develop your DApp. Furthermore, you need web3 object provided by the web3.js library to communicate with the Ethereum blockchain. Web3 communicates with any Ethereum node which exposes an RPC layer and communicates with it trough RPC calls.

INSTALLING METEOR

OSX / LINUX

Install the latest official Meteor release from your terminal:

```
$ curl https://install.meteor.com/ | sh
```

WINDOWS

Download and run the official Meteor installer:

https://install.meteor.com/windows

CREATE A METEOR PROJECT

To create the app, open your terminal and type:

```
$ meteor create sydEthereumDiploma
```

This will create a new folder called sydEthereumDiploma with all of the files that a Meteor app needs:

```
client/main.js          # a JavaScript entry point loaded on the client
client/main.html        # an HTML file that defines view templates
client/main.css         # a CSS file to define your app's styles
server/main.js          # a JavaScript entry point loaded on the server
package.json            # a control file for installing NPM packages
.meteor                 # internal Meteor files
.gitignore              # a control file for git
```

RUN THE METEOR APP

To run the newly created app:

```
$ cd sydEthereumDiploma
$ meteor
```

Open your web browser and go to http://localhost:3000

HTML FILES IN METEOR

- Meteor parses HTML files and identifies three top-level tags: <head>, <body>, and <template>.
- Templates can be used in HTML with `{{> templateName}}` or referenced in .js with `Template.templateName`
- You can add logic and data to the templates with {{}}
- Adding CSS

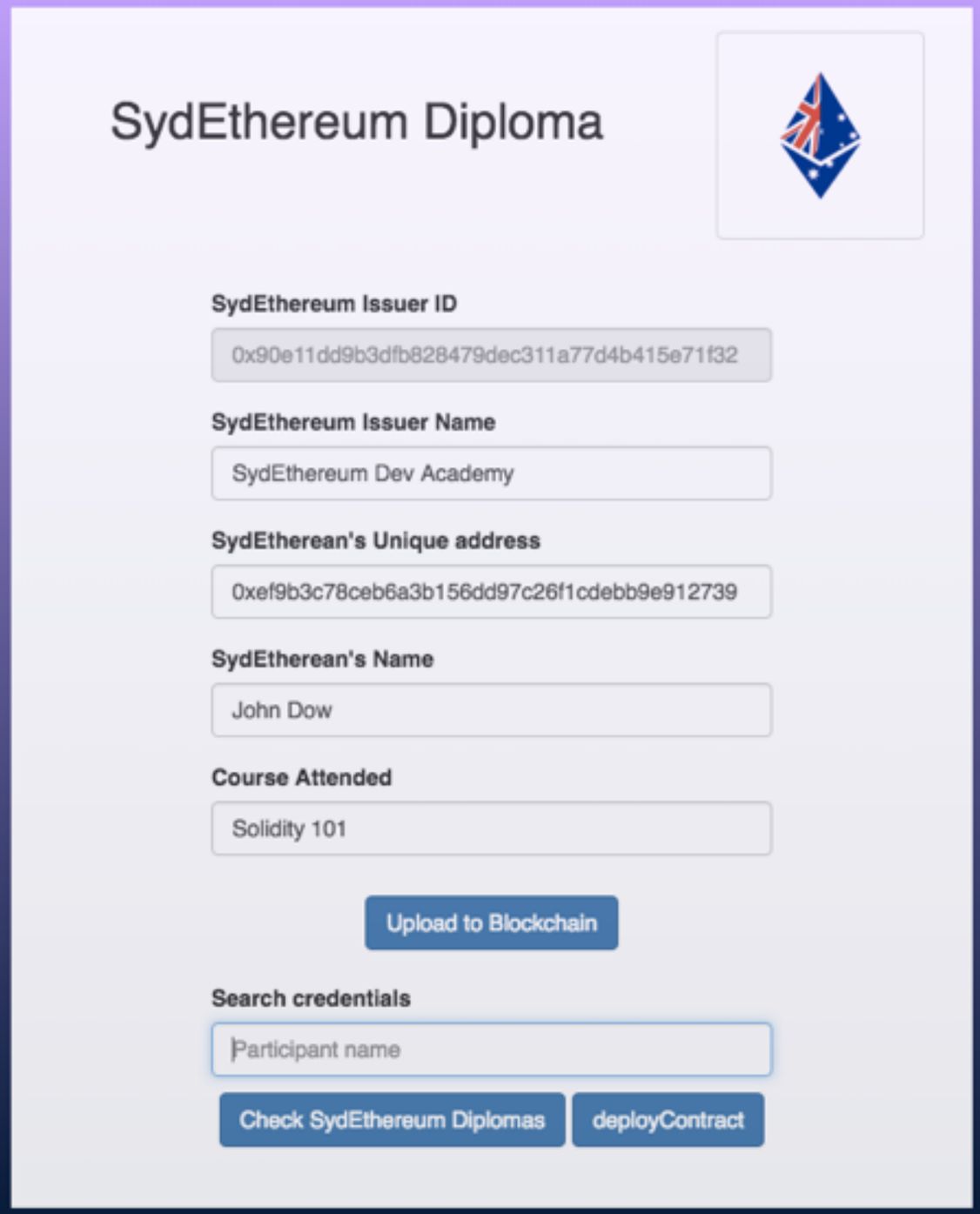
GO TO CODE >> MAIN.HTML & MAIN.JS

HELPERS AND EVENTS

- You can pass data into templates from your JavaScript code by defining helpers.
- You can add event listener to the template `Template.templateName.events(…).`
- Listen to an event that matches a specific CSS selector.
- The event handler gets an argument called `event` that has information about the event that was triggered.

GO TO CODE >> MAIN.JS

ADD HTML&CSS FILES



INSTALL WEB3

Node.js

```
$ npm install web3
```

Meteor.js

```
$ meteor add ethereum:web3
```

As Browser module

Bower

```
$ bower install web3
```

Component

```
$ component install ethereum/web3.js
```

Include ethereum.min.js in your html file. (not required for the meteor package)

SETUP WEB3

```
let Web3 = require('web3');

if (typeof web3 === 'undefined') {

    web3 = new Web3(new
    Web3.providers.HttpProvider('http://
    localhost:8545'));

   web3.eth.defaultAccount =
web3.eth.accounts[0];

}
```

GO TO CODE >> INIT.JS

ASYNCHRONOUS CALLBACKS

- Callback is a piece of executable code that is passed as an argument to other code, which is expected to call back (execute) the argument at some convenient time. The invocation may be immediate as in a synchronous callback, or it might happen at a later time as in an asynchronous callback.
- Web3 API is designed to work with local RPC node so its functions use synchronous HTTP requests by default. To make an asynchronous request pass an optional callback as the last parameter.
- Web3.js provides callbacks for async. requests so you don’t have to wait for a transactions to complete to do stuff in the front-end.
- You can use promises for better-looking code.
- Why we need them in this case?

INVOKE A TRANSACTION & GET TRANSACTION DATA

- web3 contains the eth object - web3.eth (for specifically Ethereum blockchain interactions) and the shh object - web3.shh (for Whisper interaction);
- A note on big numbers in web3.js;
- Web3.js API Reference https://github.com/ethereum/wiki/wiki/JavaScript-API
- GO TO CODE>> MAIN.JS

LISTENING FOR EVENTS

- Events allow the convenient usage of the EVM logging facilities. Events are inheritable members of contracts. When they are called, they cause the arguments to be stored in the transaction's log.
- Front-end subscribes to events via Web3.
- GO TO CODE>> SydEth.sol & Main.js

USED MATERIALS:

- https://www.meteor.com/
- https://github.com/ethereum/wiki/wiki/JavaScript-API
- https://solidity.readthedocs.io/en/latest/
- https://medium.com/@ConsenSys/a-101-noob-intro-to-programming-smart-contracts-on-ethereum-695d15c1dab4#.2es65dv3s