INSTRUCTOR LED WEBINAR - SYLLABUS

ETHEREUM DEVELOPER

Duration:	16 Hours
Delivery:	Live Instructor Led Webinar - 16 Hours
Instructor(s):	TBD
Office Hours:	10:00 AM to 6:00 PM Eastern Standard Time
Email:	studentsupport@blockchainhub360.com
Prerequisites:	Minimum 1 year software development experience. C++, Python, and Java Scripts - Ethereum's built in programming language Solidity was influenced by C++, Python and Javascript; enrolling students need to be familiar with these languages to program in Solidity

Continuing Education Units: 1.6

Certification Exam:	Certified Ethereum Developer
Certification Body:	Blockchain Certification Association

Course Overview:
The Ethereum Developer course is designed for developers who wish to understand how to create and integrate Ethereum based apps. The course gives a top down view of all the important aspects in creating such an app, an overview of how the Ethereum blockchain works, how it is different from traditional architecture, and how to setup a working environment comprised of new developer tools. Students are introduced to Solidity, the main language used for creating smart contracts. After learning how to write, deploy, test and interact with smart contracts students see some real examples in action by mimicking real running protocols. Graduating students will connect smart contracts to a user interface illustrating their ability to develop sophisticated dApps (Decentralized Applications).

Course Composition:

Instructor Led Webinar:	Ethereum Developer	Modules 1 - 4

Learning Objectives:
- Configure a Ethereum node – testRPC/parity
- Create a working environment for deploying and interacting with smart contracts
- Read and write smart contracts written in Solidity
- Illustrate how ERC20/223 tokens work, create tokens and offer them to the public with an ICO
- Develop a token exchange with Ethereum using multiple different approaches
- Deploy Multisig wallets
- Design and construct a user interface that can work with Ethereum based applications

Demonstration of Learning Outcomes:
At the conclusion of the Ethereum Developer course developers have the ability to read and write smart contracts, deploy wallets and a working blockchain dApp (distributed application).

Evaluation:
Evaluation is based on participation and a final exam.
Weighted:
50% participation
50% on the final grade
80% overall grade is required in order to receive a Certificate of Completion.

Grading Policy:
Pass or Fail. No Credit (NC).

Attendance Requirements:
Students are expected to attend at least 70% of Instructor Led Webinar Presentations. Should a student miss any portion of the live instruction instructor led webinars are recorded and attached to the learning management. A Certificate of Completion will not be issued if attendance requirements are not met.

Student conduct and etiquette:
Students will be expected to be courteous in their conduct and communications to the instructor and classmates at all times whether such conduct or communication is in person, by telephone or electronic communications.

Behavior that persistently or grossly interferes with instructor or other student activities is considered disruptive behavior and may be subject to disciplinary action. Such behavior inhibits other students' ability to learn and an instructor's ability to teach. The instructor may require a student responsible for disruptive behavior to leave the learning environment pending discussion and resolution of the problem and may report a disruptive student to the Student Affairs Office

Note: Disruptions, or any other distraction in the learning environment may result in a failing grade.

Course Evaluations
Course evaluations and program surveys are important components of the educational process. Students are encouraged to complete the student course evaluation form issued at the conclusion of the course. The evaluation is anonymous.

Computer/Information Literacy Expectations for Students enrolled in this class
Students in this class are expected to:
1. Use a word processing program for writing assignments (e.g., Microsoft Word)
2. Be able to access assigned websites through the internet
3. Have access to PC or mobile device for participation in course content

Course Module Overview:

ETHEREUM DEVELOPER

Module 1: Basic Concepts and Configuring a Working Environment
The first lessons will cover some basic concepts in blockchains and the many tools it implements. We'll cover some of the main differences between traditional and blockchain centered architecture and we'll set our working environment and tools.

- The origin of the blockchain and its basic working mechanism
- Consensus and the blockchain. How to agree on things
- Asymmetric (key) encryption
- Transactions and scripts
- How the Ethereum VM works. Storage, transactions, OP_CODES etc
- Installing and configuring truffle framework
- Creating a private blockchain using testRPC and parity
- Working with RPC and HTTP requests
- Using truffle to deploy smart contracts and run tests
- Writing tests for our smart contracts
- Using nodeJS as the back of our app

By the end of this module the students will be able to:

- Explain the basic working mechanism of blockchain (specifically, the Ethereum blockchain and Virtual Machine).
- Configure their own nodeJS, testrpc/parity nodes, and truffle projects.
- Write simple tests for smart contracts

Module 2: INTRODUCTION TO SOLIDITY:
The main language used for creating smart contracts. In this session, we'll cover some of it's basic syntax and structure.

- The smart contract as an object on the blockchain
- Variables, types, arrays, mapping, memory and storage
- Inheritance and classes. Interaction between contracts, calls and libraries
- Functions, constructors, modifiers and control flow
- In-line assembly
- Security considerations

By the end of this module the students will be able to:

- Write their own smart contracts using Solidity.
- Compile and debug their smart contracts
- Write useful test cases for their codes
- Deploy their smart contracts to the Ethereum network
- Interact with deployed smart contracts

Module 3: EXPLORING SMART CONTRACTS AND ETHEREUM PROTOCOLS:
Now that we know how to write, deploy, test and interact with smart contracts it's time to see some real example in action. All topics in this section will be covered by learning and mimicking real running protocols.

- ERC20 and ERC223 protocols
- Token issuance mechanisms
- Identity management systems (uport, civic etc.)
- Token exchanges
- Wallets and multisig

By the end of this module the students will be able to:

- Create their own tokens.
- Issue tokens to the public as an ICO
- Create and interact with identity management system
- Exchange tokens in a token exchange smart contract and in offline protocols
- Deploy and use their own multisig wallet

Module 4: FINALIZING YOUR APP:
We know how Ethereum works and how to write smart contracts for it. We've also created some sophisticated apps and made sense of contracts already running on the Ethereum blockchain. Now it's time to create a user interface for our costumers/end users.

- Using web3JS library
- Connecting your app to an Ethereum node
- Architectural considerations – how and when to use the blockchain
- Building the proper framework for displaying and receiving information to/from the user

By the end of this module the students will be able to:

- Connect any smart contract to a user interface
- Demonstrate in a graphical way the working of their smart contracts

Final Exam