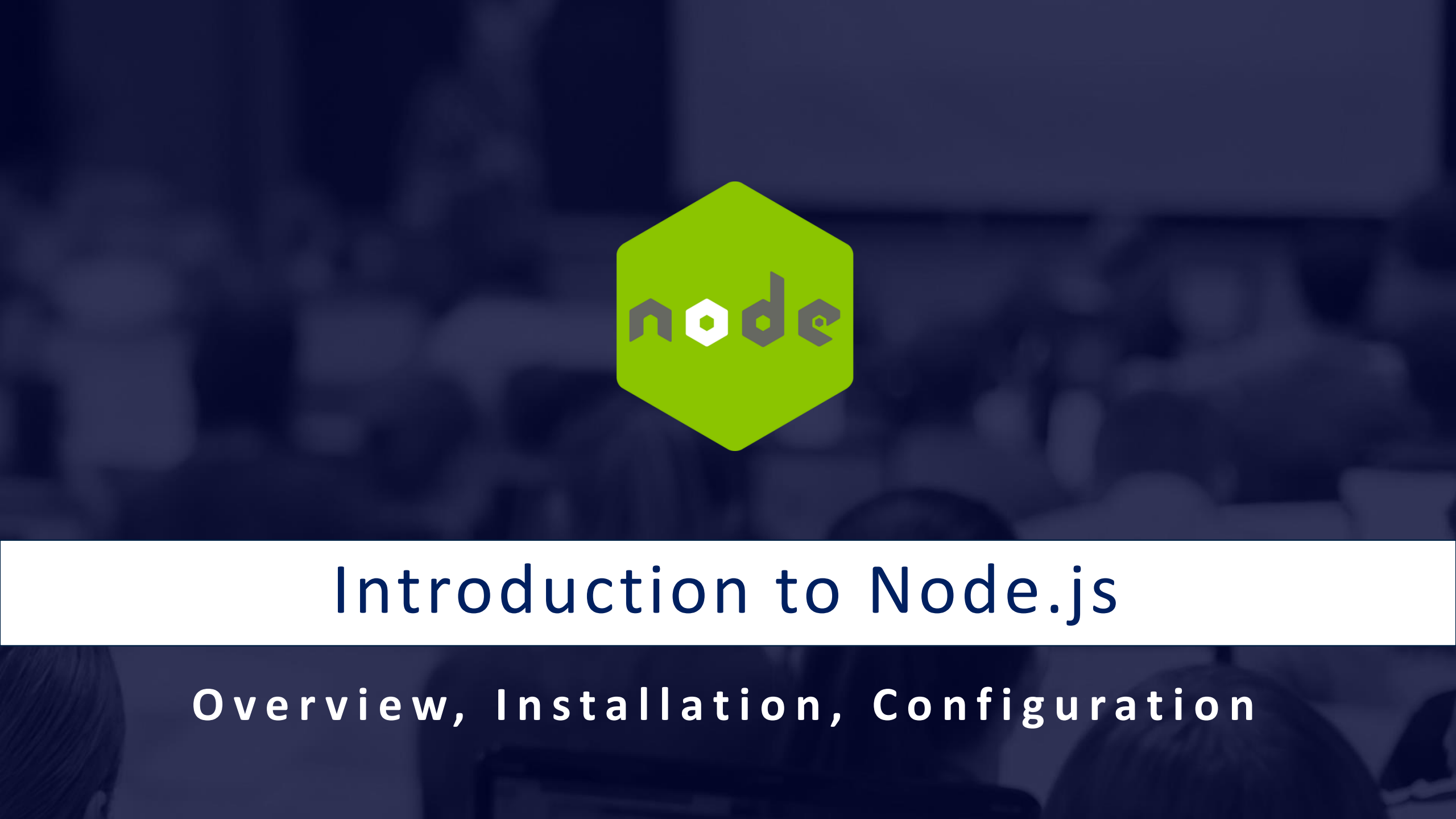
Introduction to Node.js

Overview, Modules, Web Server, Request and Response

Table of Contents



Introduction to Node.js

Overview, Installation, Configuration

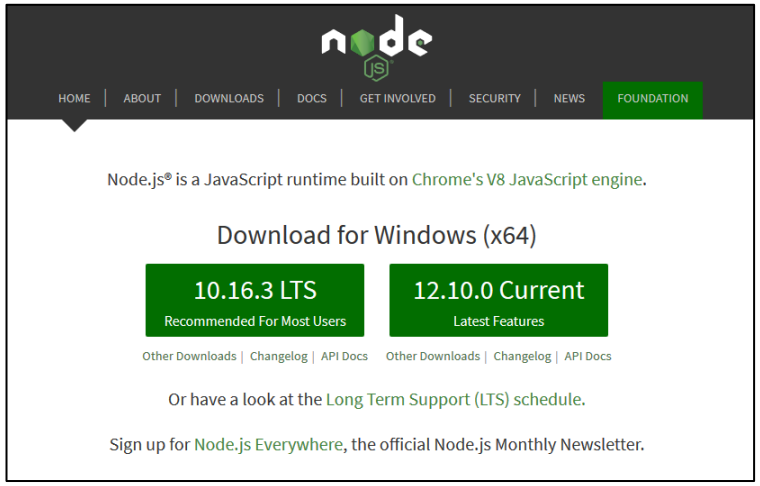
Node.js Overview

A runtime environment for JS that runs on the server

- Advantages
  - **One language** for server and client
  - **Asynchronous** and **Event** Driven
  - Very **fast**
  - Efficient **package manager**
  - No **buffering**

Installation

- Go to **http://nodejs.org** and install the latest version



- To check the currently installed version of node, type in the **command prompt** / **terminal**:

```
node -v
```

Environment Setup

- From the **terminal**

```
node      // Starts REPL
let a = 5
let b = 3
a + b     // 8
```

- Interpret code from a **file**
  - Save script to **index.js**
  - Execute from the terminal:

```
node index.js
```

NPM Packages

Node.js **projects** are usually set up as **NPM packages**

- From the **terminal**, inside the **target directory**

```
npm init
```

- Answer **questions** to initialize project
- A **`package.json`** file will be created with initial configuration
- To bypass all questions (take default values):

```
npm init -y
```

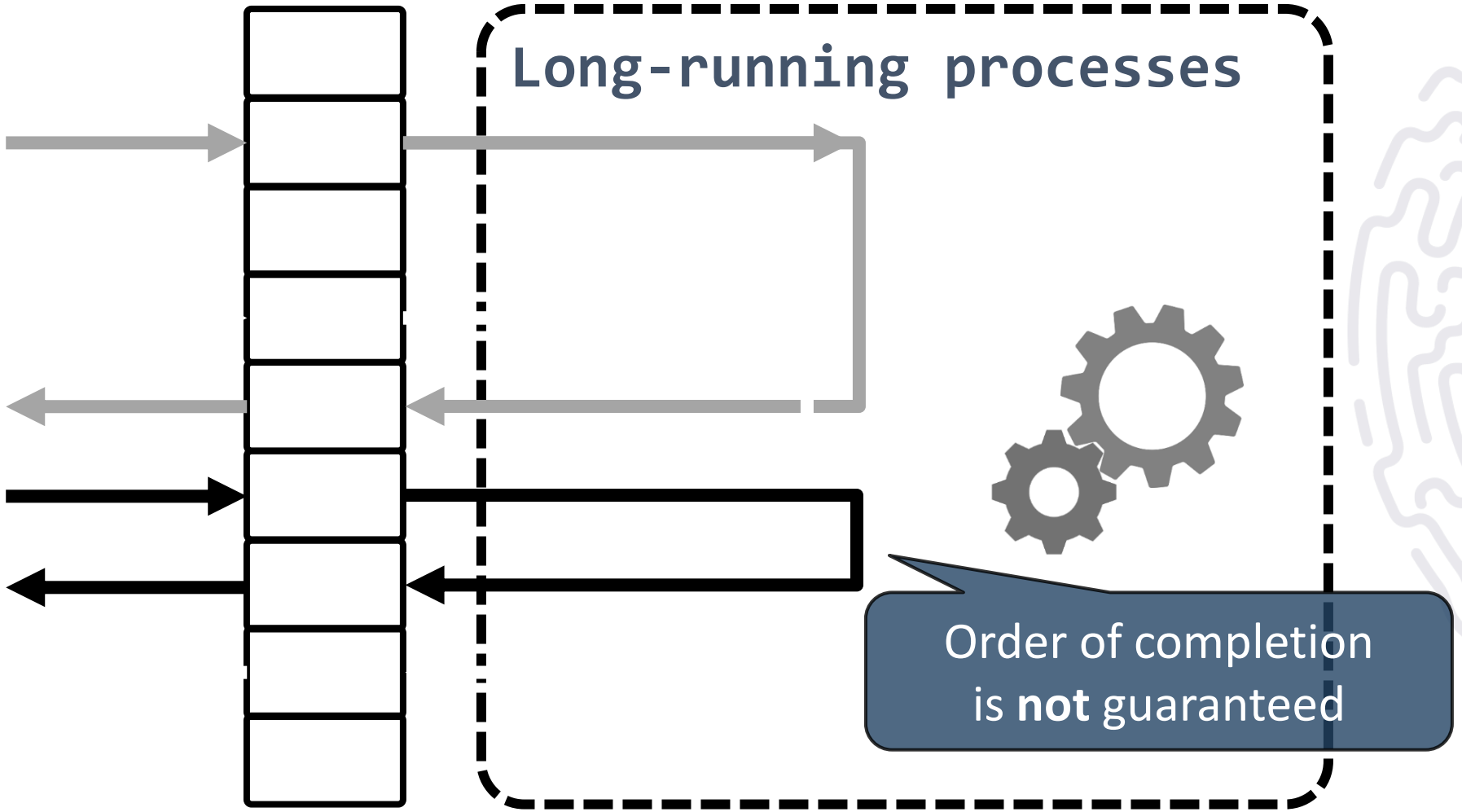
Configuration (Package.json)

```
{
  "name": "demo",
  "version": "1.0.0",
  "description": "Node.js demo project",
  "main": "index.js",
  "engines": {                    // Sets versions of Node.js
    "node": ">= 6.0.0",              and other commands
    "npm": ">= 3.0.0" },
  "scripts": {                  // Defines a set of node scripts
    "start": "node index.js" },
  "keywords": [],
  "author": "",
  "license": "ISC“
}
```

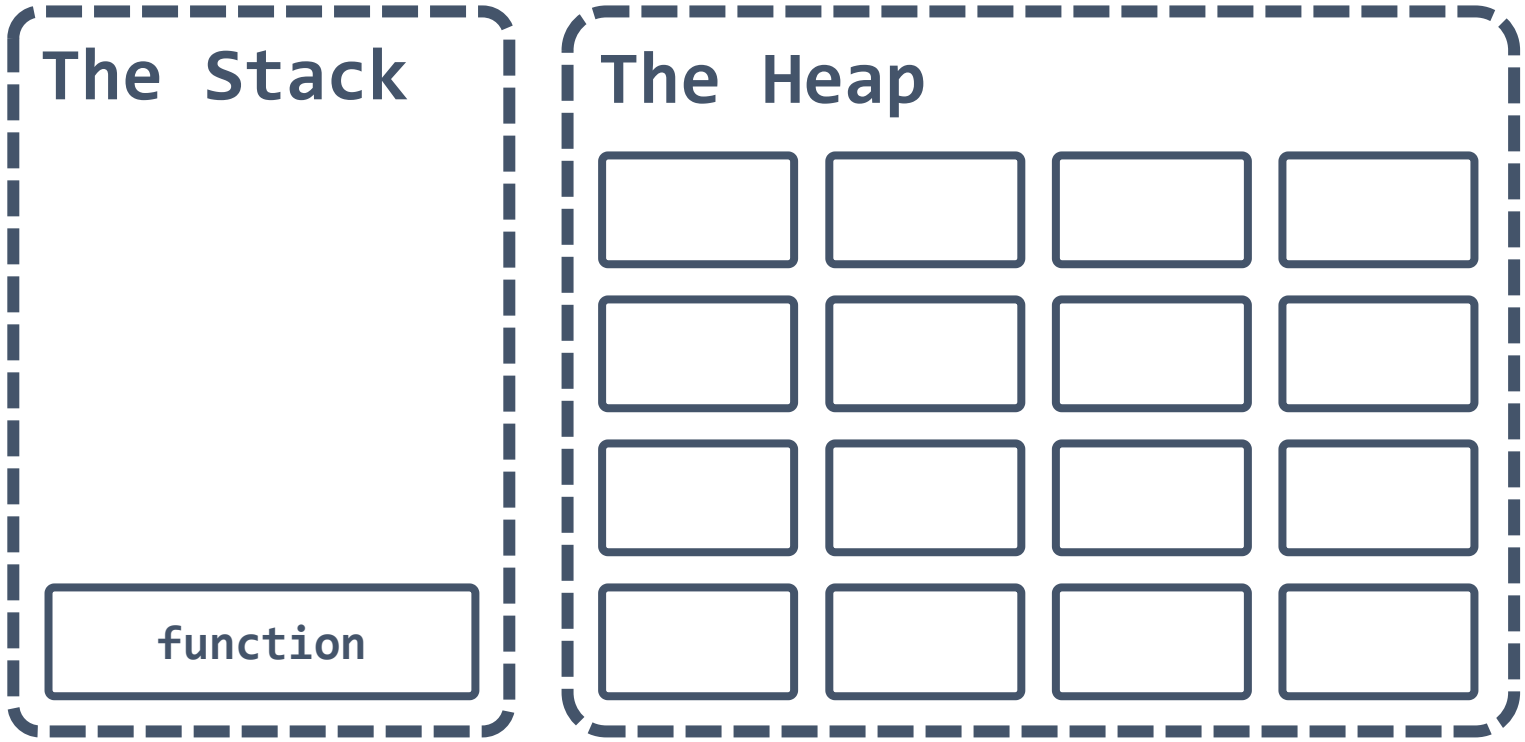
Event Loop

Event-Driven Programming

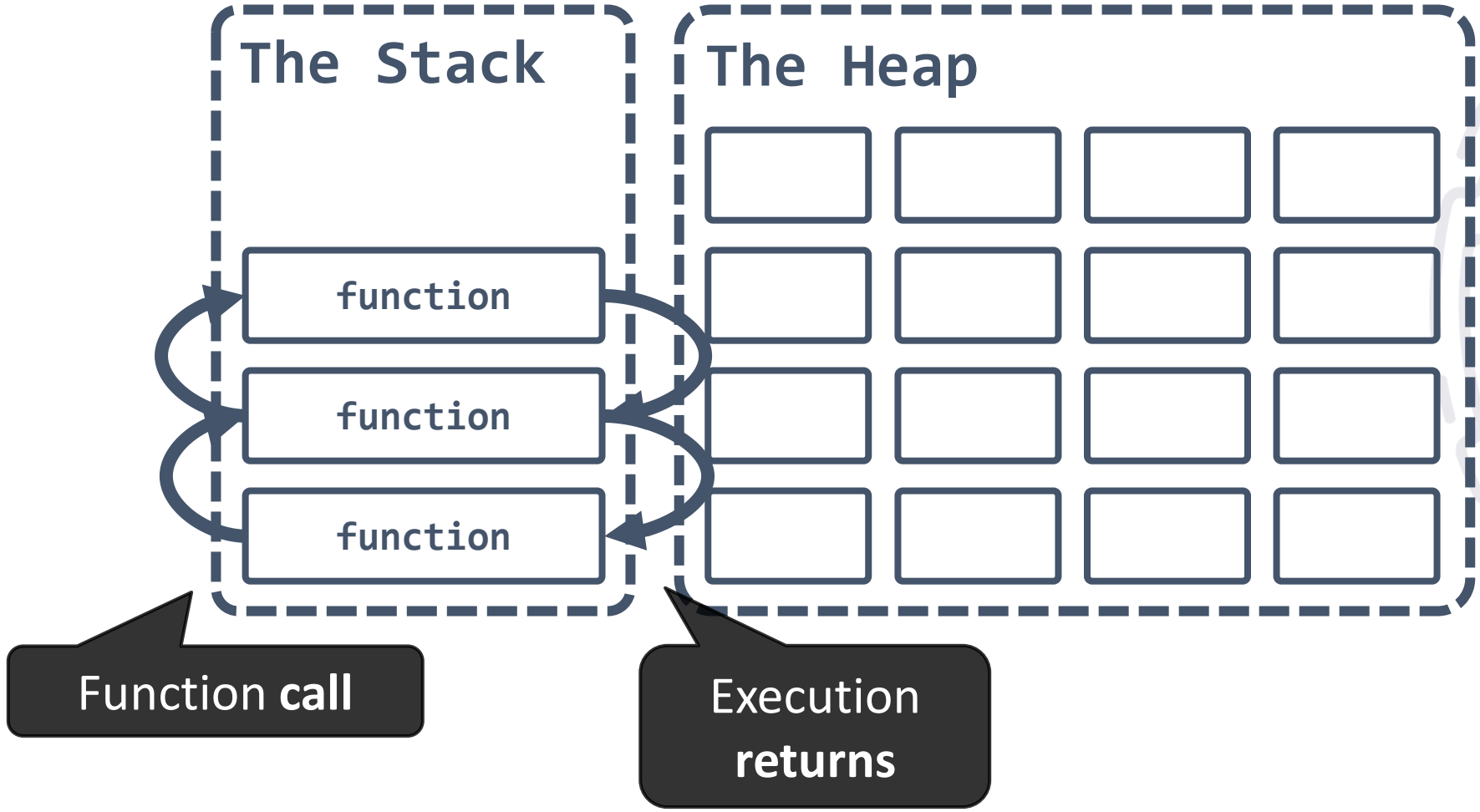
The Event Loop



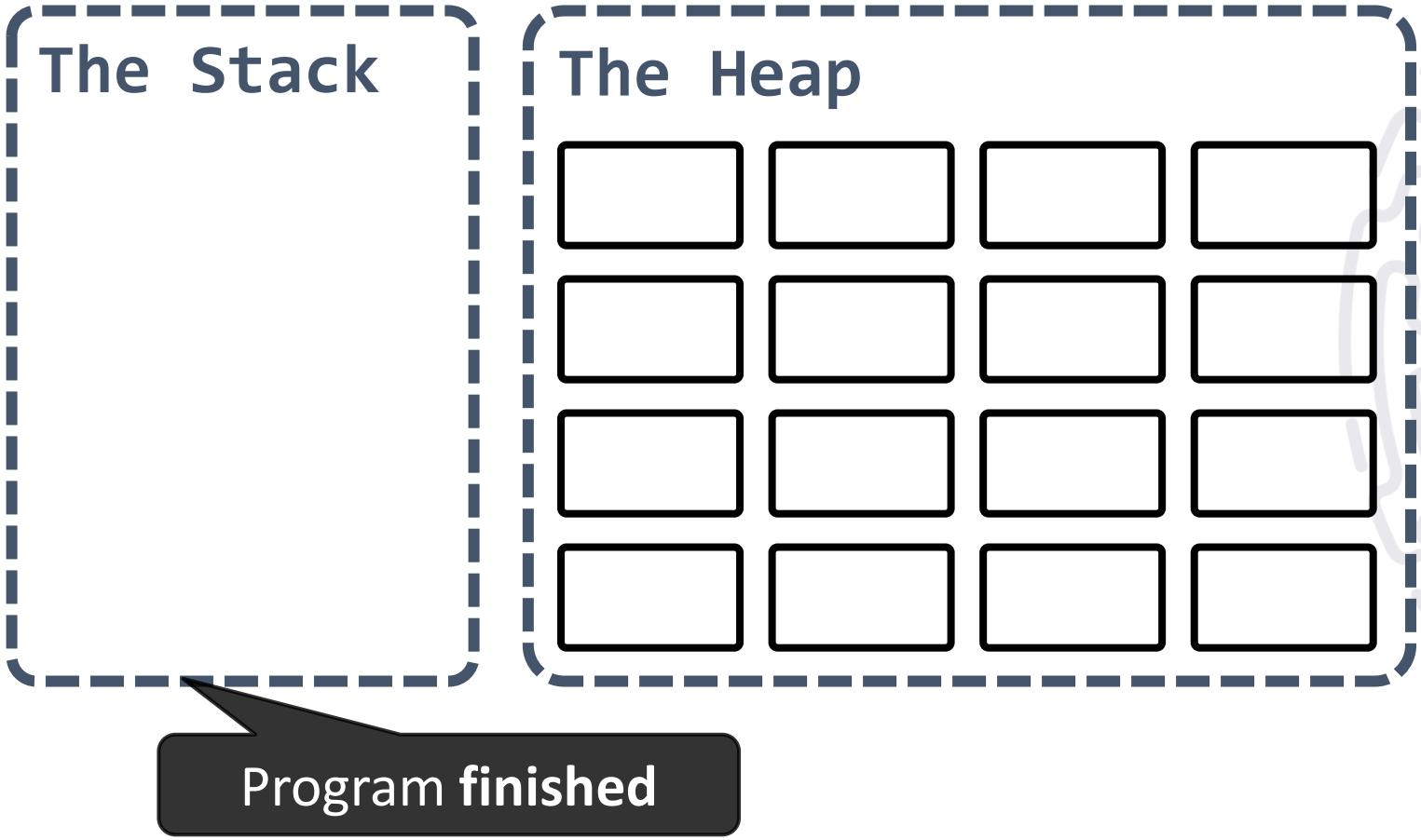
Stack Execution



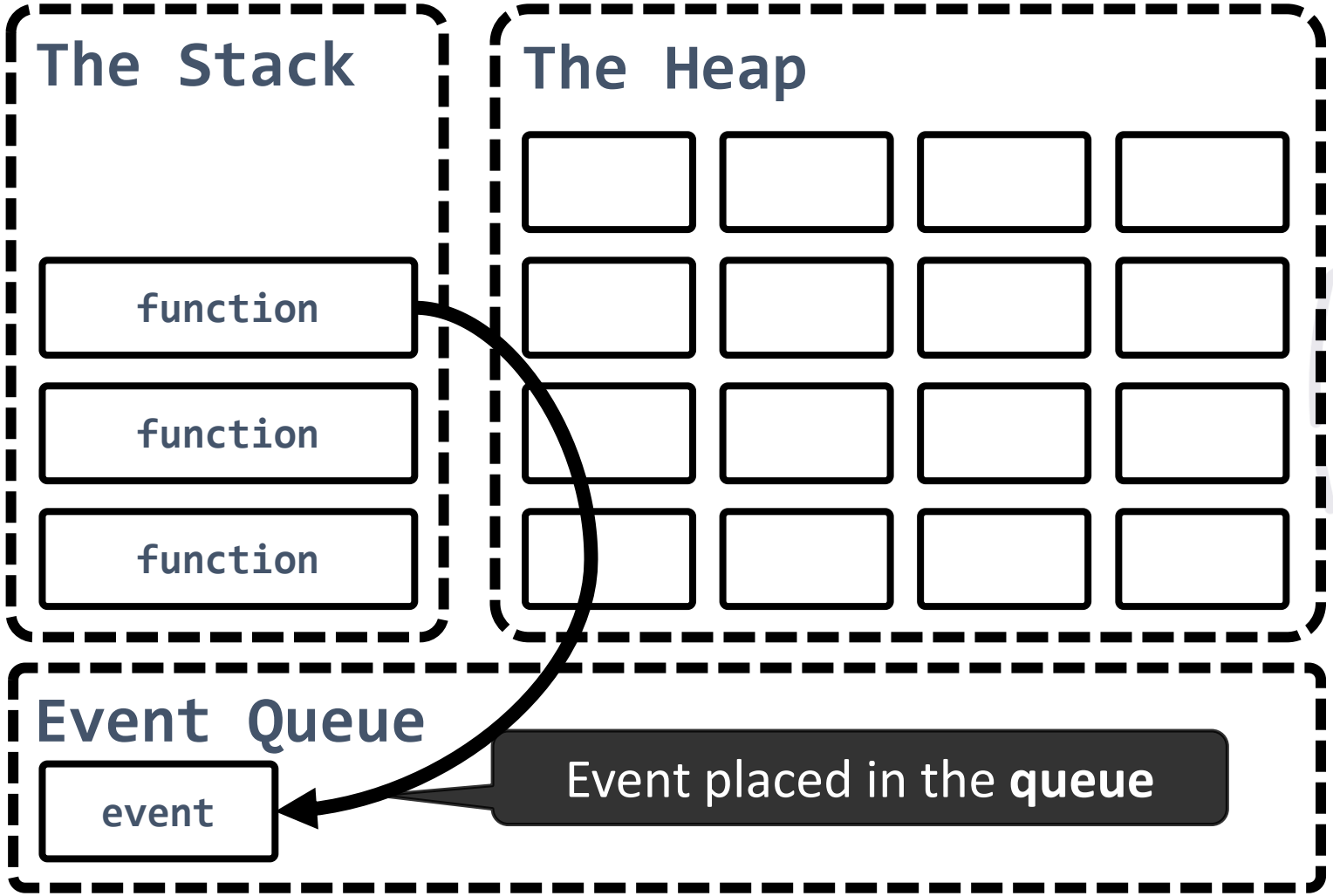
Stack Execution



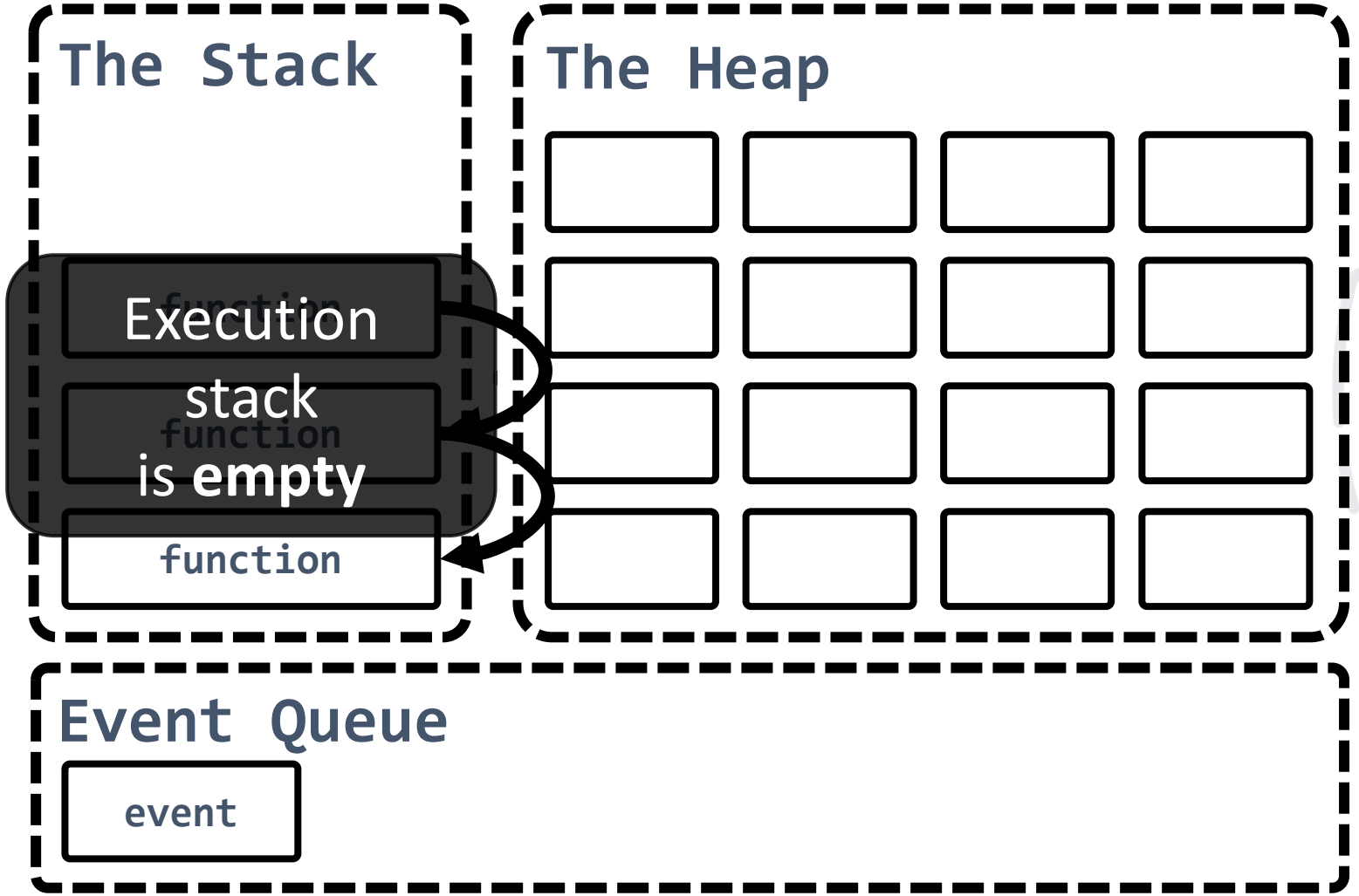
Stack Execution



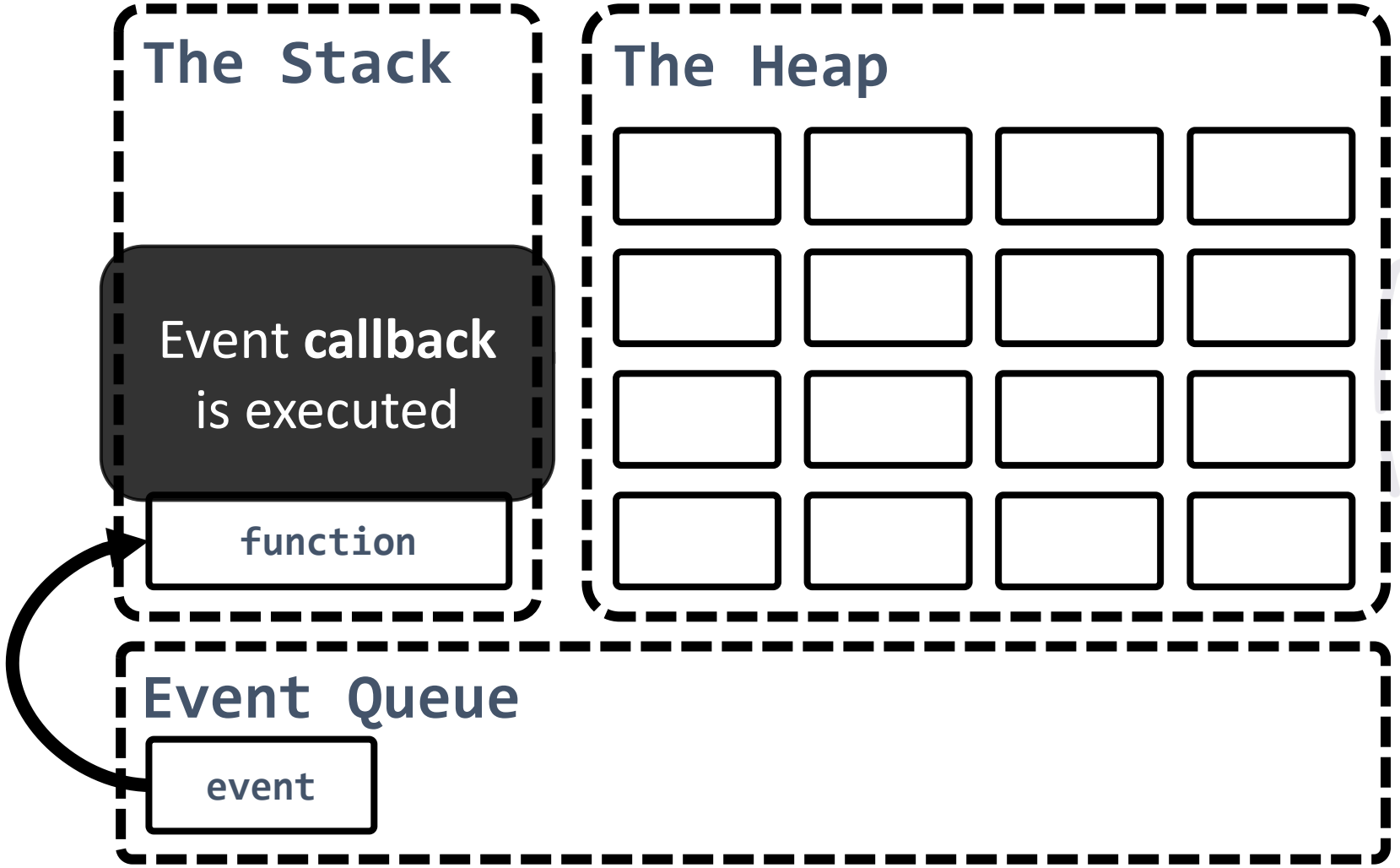
The Event Loop



The Event Loop



The Event Loop



Modules

Making Modular App

Modules

- Allow larger **apps** to be **split** and **organized**
- Each module has its **own context**
  - It **cannot pollute** the **global scope**
- Node.js includes **three types** of modules
  - **Core** Modules
  - **Local** Modules
  - **Third Party** Modules

Local Modules

- Created **locally** in the Node.js application
- Include **different functionalities** in **separate** folders
- Use **`module.exports`** to expose a **function**, **object** or **variable**

```
module.exports = myModule
```

- Loaded using the **`require()`** function

```
const myModule = require('./myModule.js');
```

Third-Party Modules

- Installed from Node Package Manager (**NPM**)
- Run from the terminal

```
npm install --save express --save-exact
```

- To use in your code

```
const express = require('express');
```

- To install globally (for use from the terminal)

```
npm install --g mocha
```

Core Modules

- Include **bare minimum functionalities** of Node.js
- Load **automatically** when Node.js process starts
- Need to be **imported** in order to be used

```
const module = require('module');
```

- Commonly used modules are
  - **http** - used to create Node.js server
  - **url**, **querystring**, **path**, **fs**

URL Module

- Provides utilities for URL **resolution** and **parsing**

```
const url = require('url');
```

- Parses an address with the **`parse()`** function
  - Returns an **object** with **info** about the **url**

```
let urlObj = url.parse(req.url);
```

- **Splits** web address into **readable** parts

URL Parts

- Host '**localhost:8080**'

```
let host = urlObj.host
```

- Path '**/home**'

```
let path = urlObj.pathname
```

- Search/query '**?year=2017&month=february**'

```
let query = urlObj.query
```

```
let search = urlObj.search
```

Query String Module

- Provides utilities for **parsing** and **formatting** URL query strings

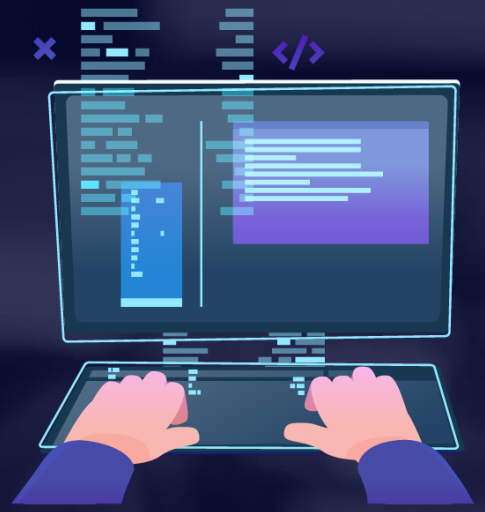
```
const queryString = require('querystring');
```

- Parses a query string into an object

```
const qs = querystring
  .parse('year=2017&month=february');
```

```
const year = qs.year;          // 2017
```

```
const month = qs.month;        // february
```



Node.js Web Server

Introduction to Web Servers

Web Servers

- All **physical** servers have **hardware**
- The hardware is controlled by the **operating system**
- **Web servers** are **software** products that use the operating system to **handle web requests**
  - Web servers **serve** Web content
- The requests are **redirected to other software** products (ASP.NET, PHP, etc.), depending on the web server **settings**

Node.js Web Server

- Creating a simple Node.js web server

```
const http = require('http');

http.createServer((req, res) => {
  res.write('Hi!');
  res.end();
}).listen(1337);

console.log('Node.js server running on port 1337');
```

Request & Response Wrappers

Handling Requests and Responses

The Request Wrapper

Used to **handle** incoming http requests

- Properties
  - **httpVersion** - '1.1' or '1.0'
  - **headers** - object for request headers
  - **method** - 'GET', 'POST', etc.
  - **url** - the URL of the request

30

Request Wrapper Example

```
const http = require('http');
const url = require('url');
const port = 1337;

http.createServer((req, res) => {
  let path = url.parse(req['url']).pathname;
  if (path === '/') {
     // TODO: Send 'Welcome to home page!'
  }
}).listen(port);
```

The Response Wrapper

Used to **retrieve** a **response** to the **client**

- Functions
  - Create **response header**
  - Send the actual **content** to the **client**
  - **End** the response

Response Wrapper Example

```
const http = require('http');
const port = 3000;

http.createServer((req, res) => {
  res.writeHead(200, { // Response Status Code
    'Content-Type': 'text/plain'
  });
  res.write('Hello from Node.js'); // UTF-8 Encoding
  res.end(); // Always End the Response
}).listen(port);
```

Live Exercises

Summary

- Node.js is a **fast** and **asynchronous** efficient **package manager**
- Applications can be **organized** using **module**
- NPM allows quick access to **external modules**
- **Web Servers** transfer resources to the **Client**
- The **Request/Response** Wrappers

Questions?

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © Kingsland University – https://kingslanduniversity.com

KINGSLAND UNIVERSITY

THANK YOU