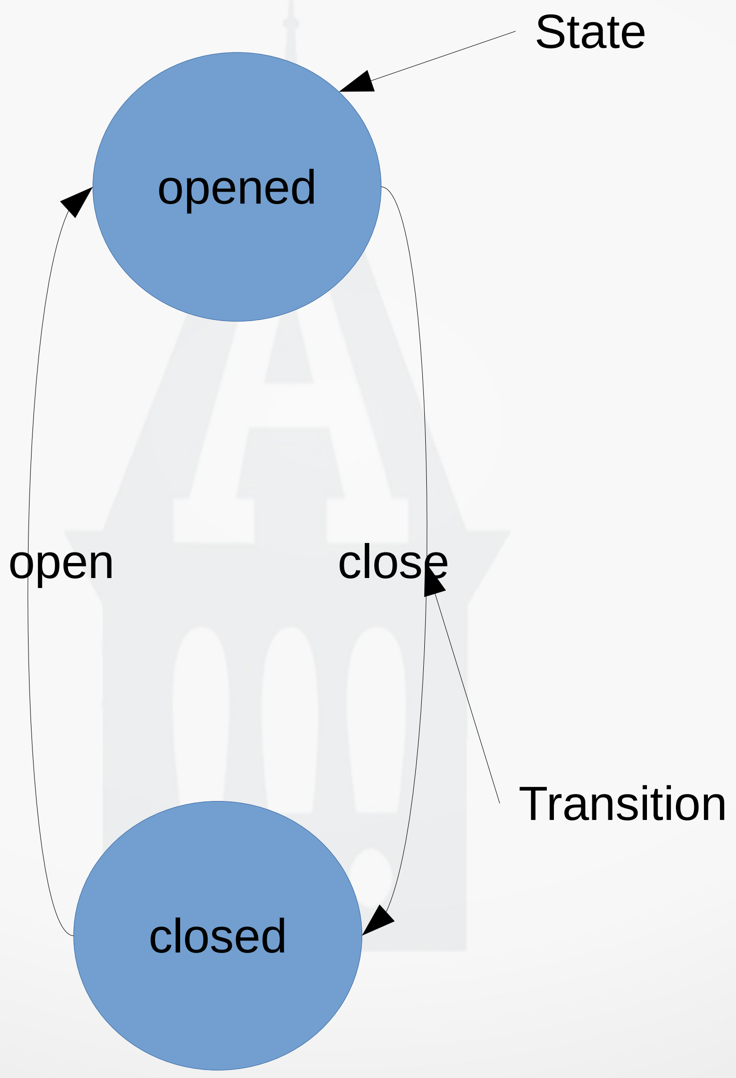
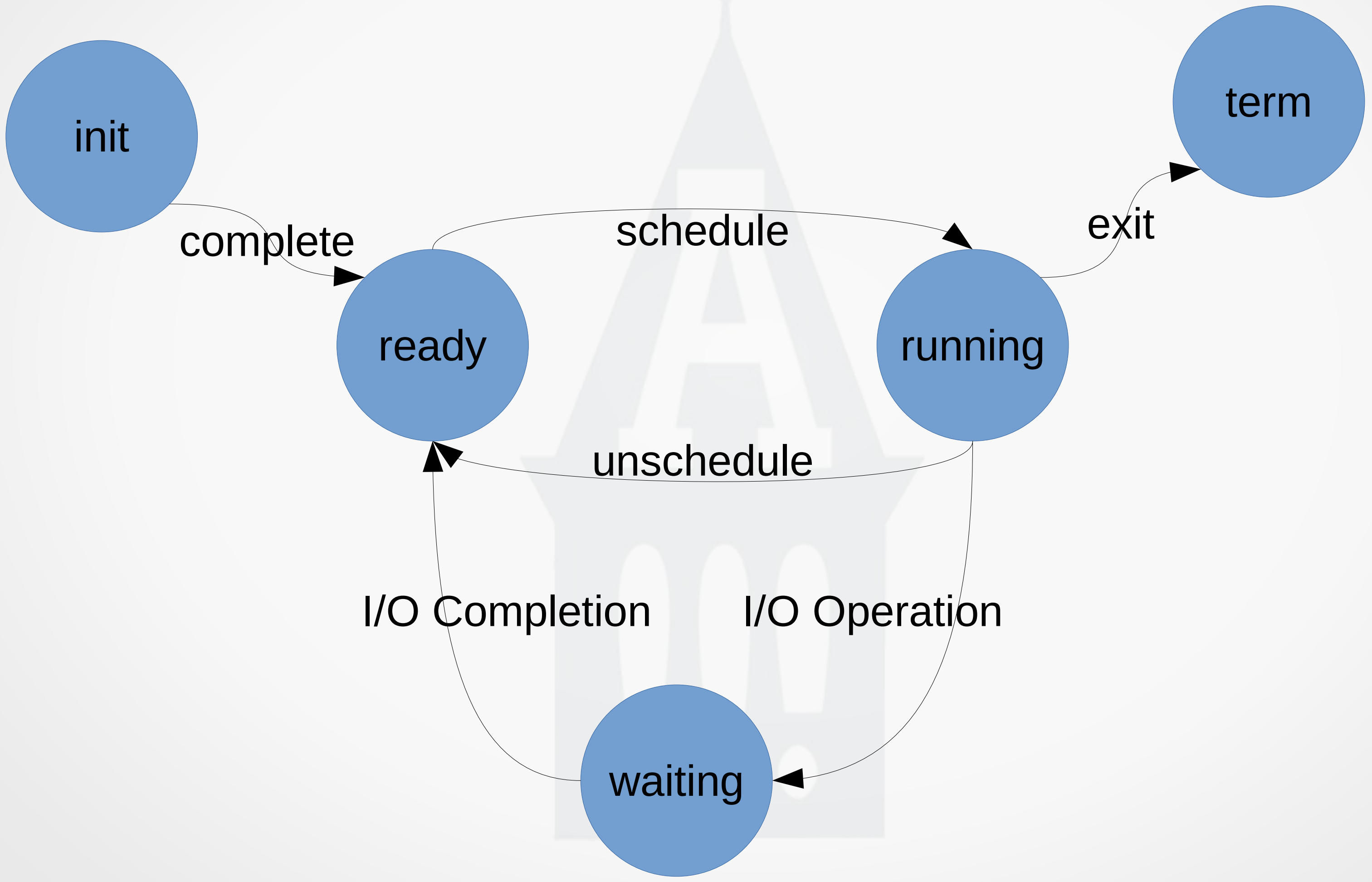
Game State Management

Internal Game & Menu State Management

Finite State Machines



Another Example



State Transition Table

State/Input	init	ready	running	waiting	term
complete	ready	...	...	...	...
schedule	...	running	...	...	...
unschedule	...	...	ready	...	...
I/O Call	...	...	waiting	...	...
I/O Finish	...	...	...	ready	...
exit	...	...	term	...	...

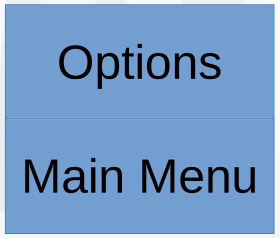
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit

Main Menu

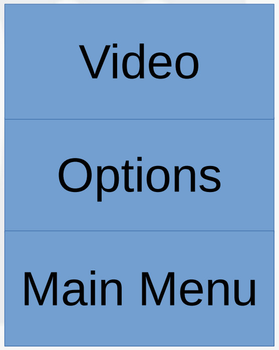
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



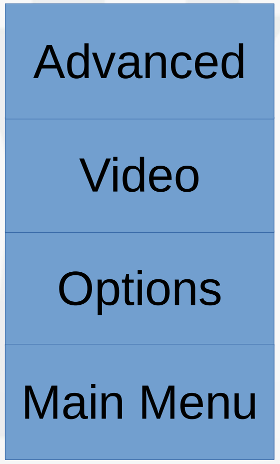
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



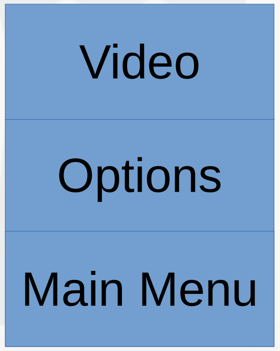
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



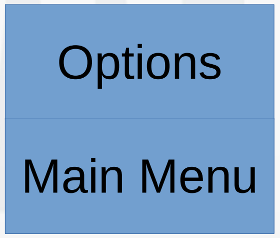
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



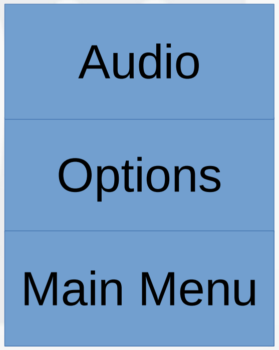
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



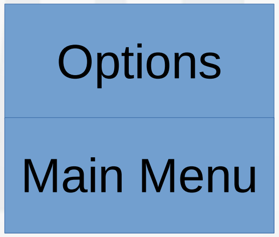
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit



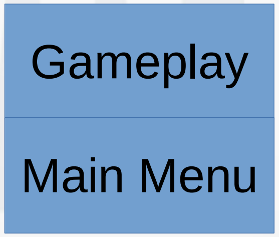
Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit

Main Menu

Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit

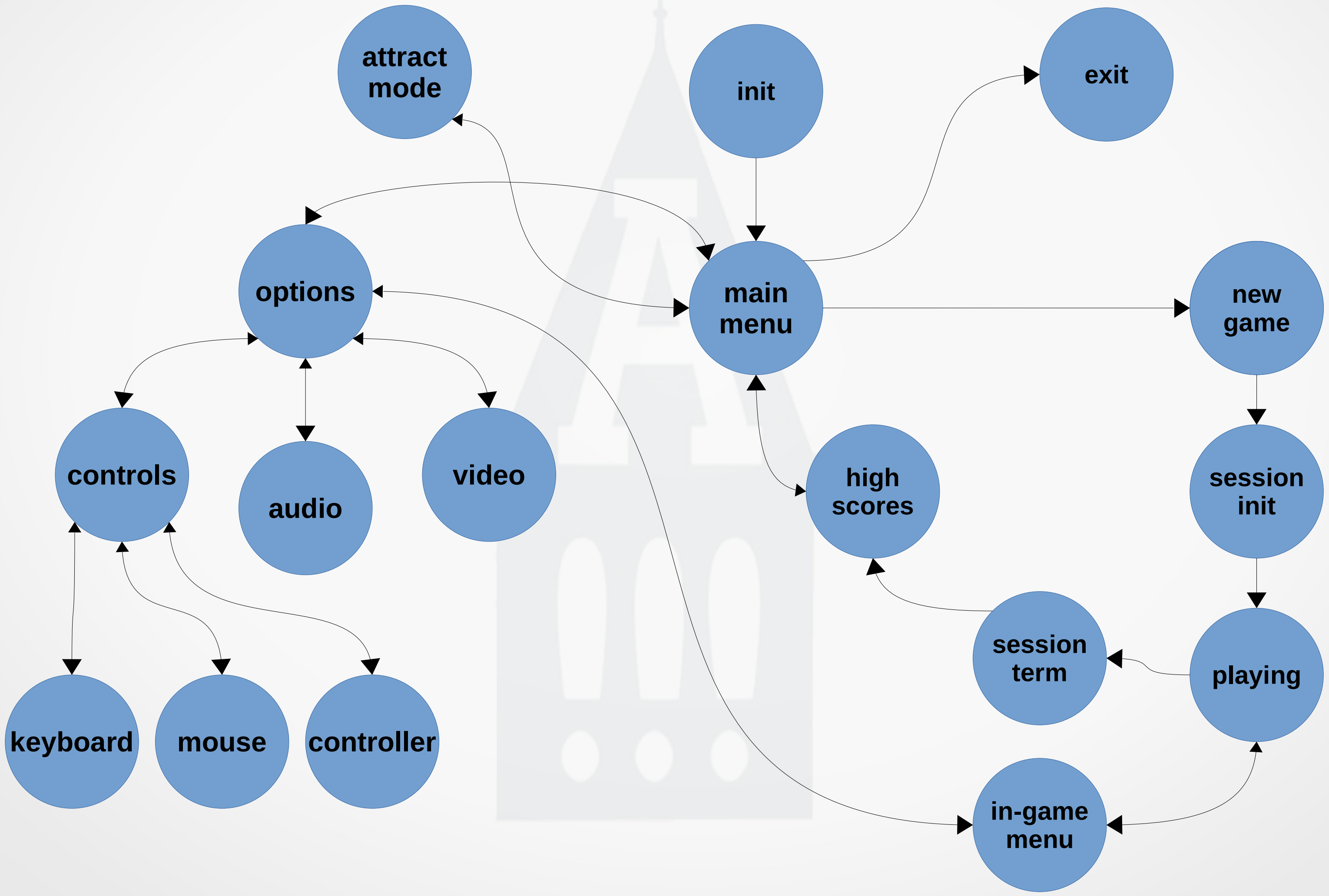


Menu State - Stacks

- New Game
- Options
  - Video
    - Basic
    - Advanced
  - Audio
  - Controls
    - Keyboard
    - Mouse
    - Controller
- High Scores
- Credits
- Exit

Main Menu

Menus – State Transition Diagram



Game State

Menuing - HTML

Game State - HTML

- Define “screens” in HTML
  - one `<div>` per menu state
  - define an ‘`id`’ per `<div>`
  - Define an “active” screen class (state) in CSS
    - active: `display: block;`
    - not active: `display: none;`
- Define “screens” in JavaScript
  - Common functional interface
    - `initialize`
    - `run`
  - Global, one-time, initialization on all “screens”
  - Global state change function: `showScreen(screen)`

Game Screens - HTML

```
<div id = "about" class = "screen">
    <h1>About</h1>
    <p>Developed by</p>
    <p>Dr. James Dean Mathias</p>
    <ul class = "menu">
        <li><button id = "id-about-back">Back</button></li>
    </ul>
</div>
```

```
<div id = "game-play" class = "screen">
    <canvas id = "id-canvas"
            width = "500" height = "500">
    </canvas>
</div>
```

Game Screens - Interface

```
MyGame.screens['game-screen'] = (function(game) {

    function initialize() {
        ... One time screen initialization code goes here ...
    }

    function run() {
        ... Active running state code goes here ...
    }

    return {
        initialize : initialize,
        run : run
    };
}(MyGame.game));
```

Game Screens - Example

```
MyGame.screens['about'] = (function(game) {
    function initialize() {
        document.getElementById('id-about-back').addEventListener(
            'click',
            function() { game.showScreen('main-menu'); });
    }

    function run() {
        ... nothing here for this state ...
    }

    return {
        initialize : initialize,
        run : run
    };
}(MyGame.game));
```

Game Screens – Load/Initialize

```
<script src = "scripts/mainmenu.js"></script>
<script src = "scripts/gameplay.js"></script>
<script src = "scripts/highscores.js"></script>
<script src = "scripts/help.js"></script>
<script src = "scripts/about.js"></script>
```

```
function initialize() {
    let screen = null;
    //
    // Go through each of the screens and tell them to initialize
    for (screen in screens) {
        if (screens.hasOwnProperty(screen)) {
            screens[screen].initialize();
        }
    }

    //
    // Make the main-menu screen the active one
    showScreen('main-menu');
}
```

Game Screens – Show Screen

```
function showScreen(id) {
    //
    // Remove the active state; there should only be one...
    let active = document.getElementsByClassName('active');
    for (let screen = 0; screen < active.length; screen++) {
        active[screen].classList.remove('active');
    }
    //
    // Tell the screen to start actively running
    screens[id].run();
    //
    // Then, set the new screen to be active
    document.getElementById(id).classList.add('active');
}
```

Game State

Inspection of Demo