CS47SI: Cross-Platform Mobile Development

Lecture 7B: Networking

James Landay
Abdallah AbuHashem
Tiffany Manuel
Cisco Vlahakis
Vy Mai

https://cs47.stanford.edu

cs47-fall19.slack.com

Fall 2019

CS 47

Cross-Platform Mobile Development

- Overview
- Schedule
- Readings

Overview

This course teaches the fundamentals of cross-platform mobile application development with a focus on the React Native framework (RN). The goal is to help students develop best practices in creating apps for both iOS and Android by using Javascript and existing web + mobile development paradigms. Students will explore the unique aspects that made RN a primary tool for mobile development within Facebook, Instagram, Walmart, Tesla, and UberEats.

COURSE LOGISTICS

Date/Time	T/Th 10:30PM - 12:00PM
Enrollment	Please apply here and show up to the first class to enroll in the class.
Location	Wallenberg 124 (160-124)
Units	2 Pass/Fail
Instructors	Abdallah Abuhashem (aabuhash@stanford.edu) Tiffany Manuel (manuel14@stanford.edu) Vy Mai (vmai2@stanford.edu) Cisco Vlahakis (vlahakis@stanford.edu)
Faculty Sponsor	James Landay (landay@stanford.edu)
Staff email	reactnative@cs.stanford.edu
Office hours	TBD
Prerequisites	CS 106A/B
Explore courses	CS47

https://cs47.stanford.edu

To access lectures use Stanford email

SWAPI

The Star Wars API

https://swapi.co

Live Exercise

Spaceship Store

Download

STARTER CODE

Promises

Not the ones people make

A Promise is a proxy for a value not necessarily known when the promise is created.

USAGE

Running asynchronous functions:

- Networking calls
- Accessing storage

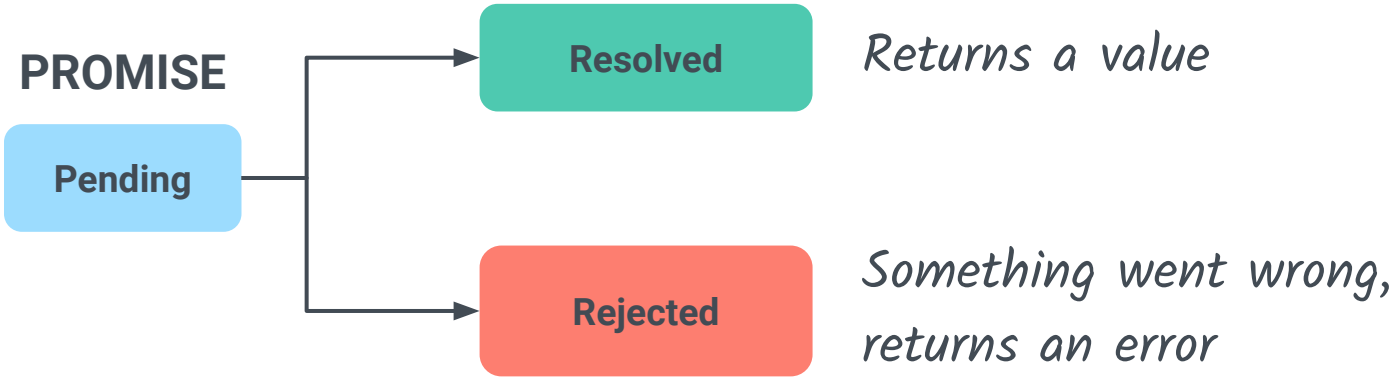
```
const promise = new Promise((resolve, reject) => {
  setTimeout(() => {
    resolve('foo');
  }, 1000);
});

promise.then(value => {
  console.log(value);
  // Expected Output: 'foo'
});
```

USEFUL LINKS / JavaScript Demo / Chaining Promises

Promises

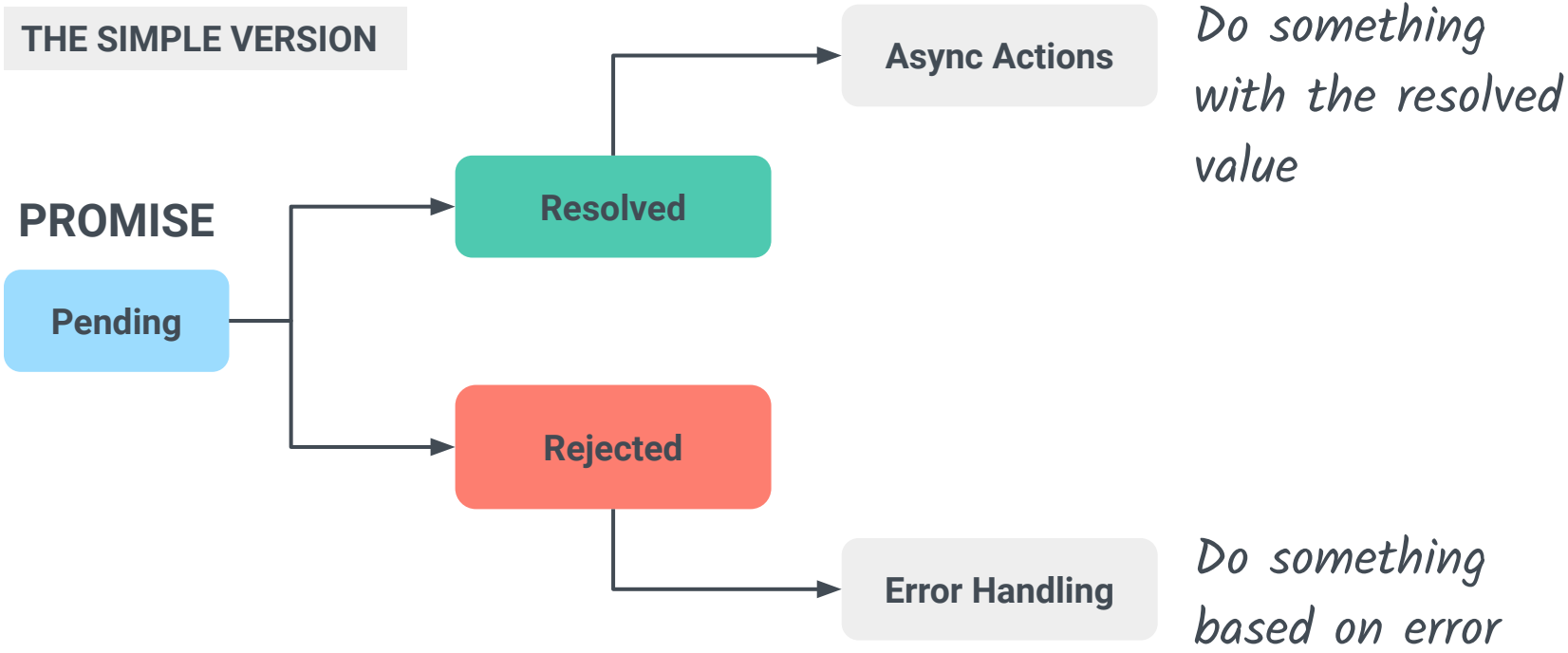
THE SIMPLE VERSION



Promises

Not the ones people make

THE SIMPLE VERSION

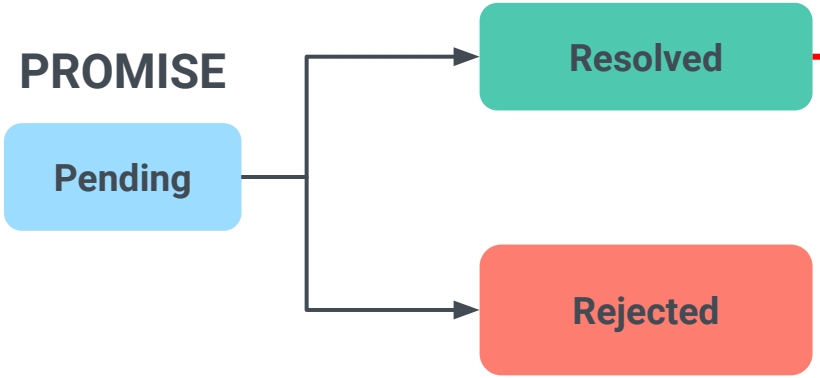


USEFUL LINKS / JavaScript Demo / Chaining Promises

Promises

Not the ones people make

THE SIMPLE VERSION



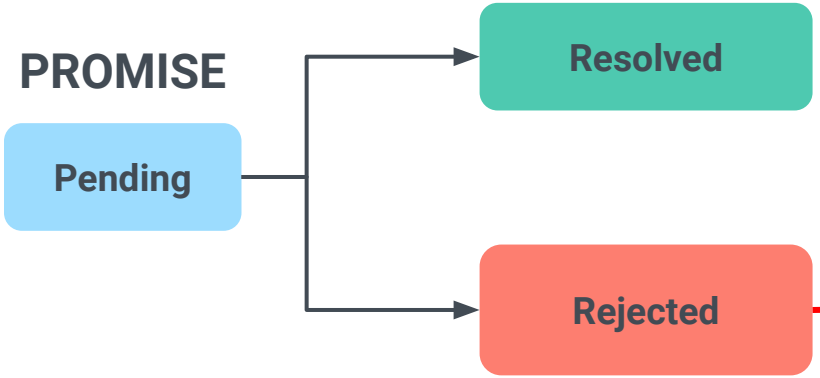

```
promise.then(value => {
  // Do something with value
}).catch(error => {
  // Do something with error
});
```

USEFUL LINKS / JavaScript Demo / Chaining Promises

Promises

Not the ones people make

THE SIMPLE VERSION



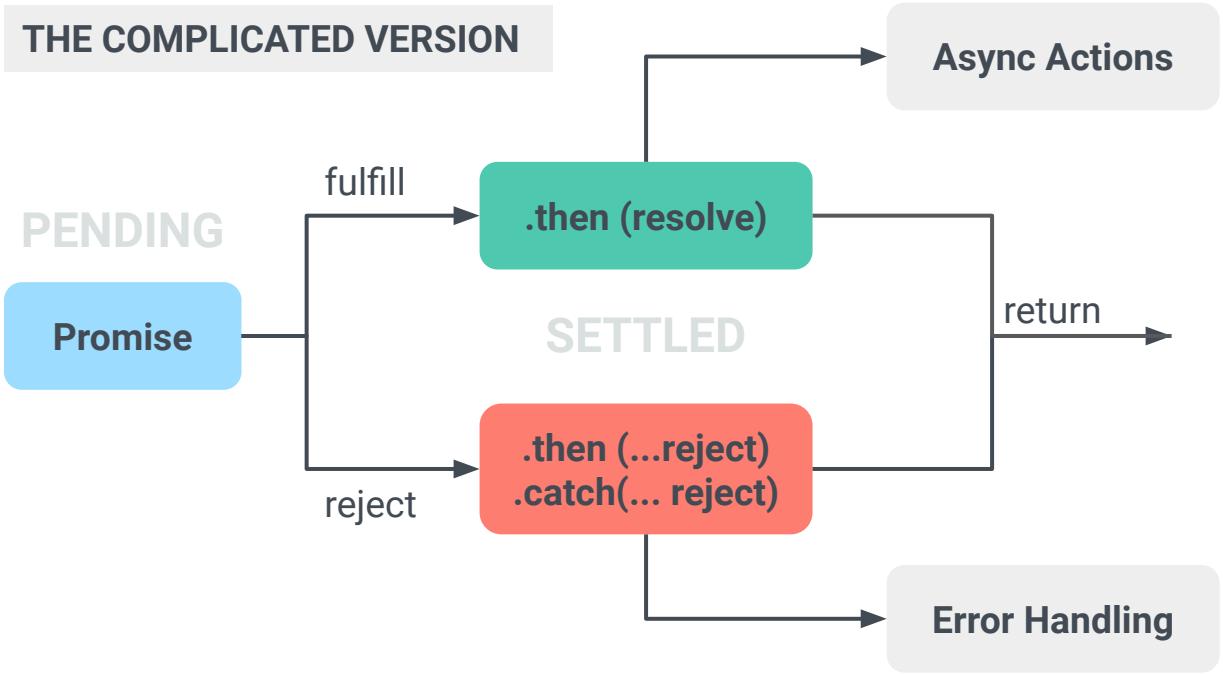

```
promise.then(value => {
  // Do something with value
}).catch(error => {
  // Do something with error
});
```

USEFUL LINKS / JavaScript Demo / Chaining Promises

Promises

Not the ones people make

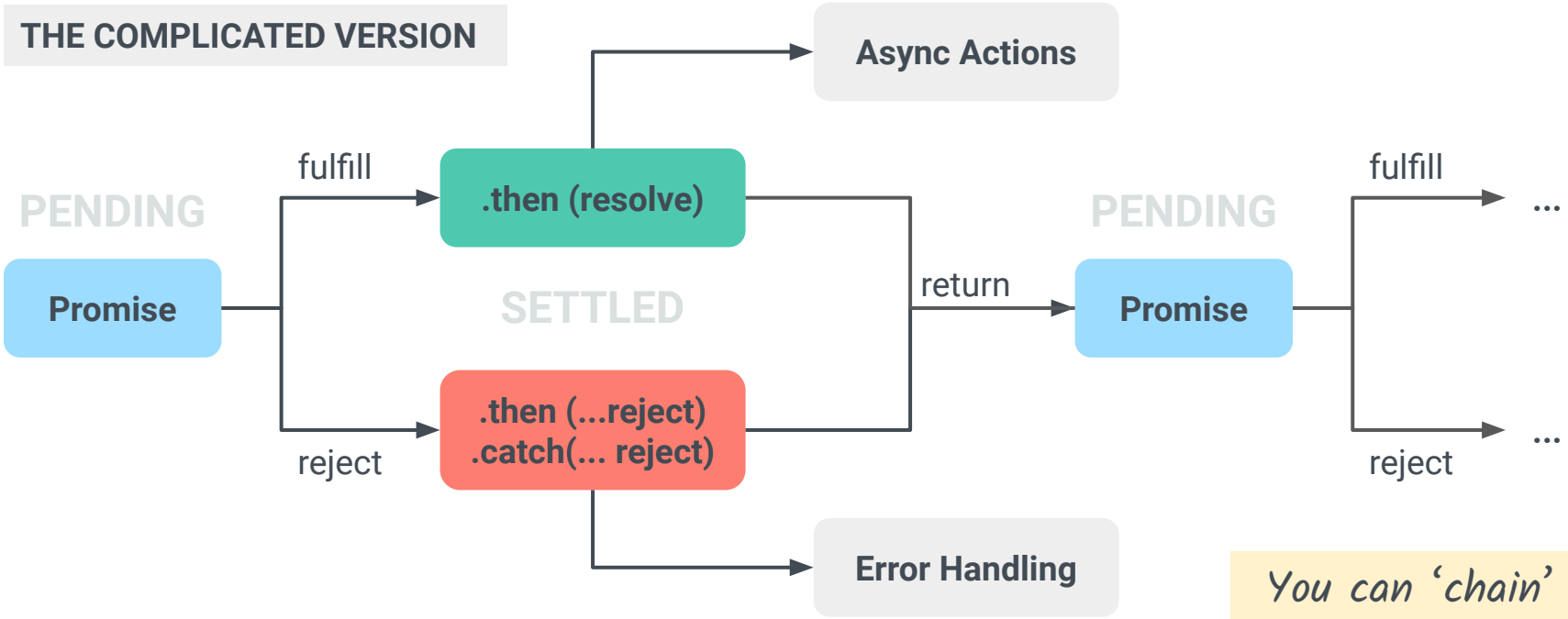
THE COMPLICATED VERSION



USEFUL LINKS / JavaScript Demo / Chaining Promises

Promises

Not the ones people make



You can 'chain' promises

USEFUL LINKS / JavaScript Demo / Chaining Promises

Chaining Promises

You can run asynchronous functions in order by chaining them.

```
const promise = new Promise((resolve, reject) => {
  setTimeout(() => resolve(1), 1000);
}).then(function(result) {
  alert(result);
  return result * 2;
}).then(function(result) {
  alert(result);
  return result * 2;
}).then(function(result) {
  alert(result);
  return result * 2;
});
```

Chaining Promises

Not the ones people make

You can run asynchronous functions in order by chaining them.

```
const promise = new Promise((resolve, reject) => {
  setTimeout(() => resolve(1), 1000);
}).then(function(result) {
  alert(result);
  return result * 2;
}).then(function(result) {
  alert(result);
  return result * 2;
}).then(function(result) {
  alert(result);
  return result * 2;
});
```

CAVEAT

Pyramid of doom

```
                                                    });
                                                 });
                                              });
                                           });
                                        });
                                     });
                                  });
                               });
                            });
                         });
                      });
                   });
```

Promises

```
myfunction = () => {
  ...
  promise1.then(value1 => {
    // Do something with value
    promise2.then(value2 => {
      // Do something with value
      promise3.then(value3 => {
        // Do something with value
      }).catch(error3 => {
        // Do something with error
      })
    }).catch(error2 => {
      // Do something with error
    })
  }).catch(error1 => {
    // Do something with error
  });
  ...
}
```

Don't do this

Promises

```
myfunction = () => {
  ...
  promise1.then(value1 => {
    // Do something with value
    return promise2;
  }).then(value2 => {
    // Do something with value
    return promise3;
  }).then(value3 => {
    // Do something with value
  }).catch(error => {
    // Do something with error
    error;
  })
  ...
}
```

A little better

Async / Await

Promises

```
myfunction = () => {
  ...
  promise.then(value => {
    // Do something with value
  }).catch(error => {
    // Do something with error
  });
  ...
}
```

Async/Await

```
myfunction = async () => {
  ...
  try {
    const value = await promise;
    // Do something with value
  } catch(error) {
    // Do something with error
  }
  ...
}
```

USEFUL LINKS / Async / Await Guide / Try...Catch Statement

Promises

```javascript
myfunction = () => {
  ...
  promise1.then(value1 => {
    // Do something with value
    return promise2;
  }).then(value2 => {
    // Do something with value
    return promise3;
  }).then(value3 => {
    // Do something with value
  }).catch(error => {
    // Do something with error
    error;
  })
  ...
}
```

Async/Await

```javascript
myfunction = async () => {
  ...
  try {
    const value1 = await promise1;
    const value2 = await promise2;
    const value3 = await promise3;
    // Do something with values
  } catch(error) {
    // Do something with error
  }
  ...
}
```

Promises

Hooray for async/await!

```
myfunction = () => {
  ...
  promise1.then(value1 => {
    // Do something with value
    return promise2;
  }).then(value2 => {
    // Do something with value
    return promise3;
  }).then(value3 => {
    // Do something with value
  }).catch(error => {
    // Do something with error
    error;
  })
  ...
}
```

Async/Await

```
myfunction = async () => {
  ...
  try {
    const value1 = await promise1;
    const value2 = await promise2;
    const value3 = await promise3;
    // Do something with values
  } catch(error) {
    // Do something with error
  }
  ...
}
```

USEFUL LINKS / JavaScript Demo / Chaining Promises

Promises

Hooray for async/await!

```
myfunction = () => {
  ...
  promise1.then(value1 => {
    // Do something with value
    return promise2;
  }).then(value2 => {
    // Do something with value
    return promise3;
  }).then(value3 => {
    // Do something with value
  }).catch(error => {
    // Do something with error
    error;
  })
  ...
}
```

Async/Await

```
myfunction = async () => {
  ...
  try {
    const value1 = await promise1;
    const value2 = await promise2;
    const value3 = await promise3;
    // Do something with values
  } catch(error) {
    // Do something with error
  }
  ...
}
```

USEFUL LINKS / JavaScript Demo / Chaining Promises

Networking

Fetch

```
const response = await fetch('https://swapi.co/api/people/');
const responseJson = await response.json();
```

SWAPI

The Star Wars API

https://swapi.co

Follow Along

https://reflect.sh/various-ocean

Live Exercise

Spaceship Store

1. Familiarize yourself with the App file structure.
2. Find the function that loads the starship data from SWAPI.
3. Use `fetch` to load data.

Ending Exercise

Load data from another API

Download

STARTER CODE

Spotify

Example

Unsplash

Map

Next Week

Firebase

Office Hours

Email us directly if you're not available at these times

Abdallah AbuHashem
Monday (12-1 PM) @ Huang Basement
Or by appointment

Vy Mai
Tuesday (3-4 PM) @ Old Union
Or by appointment

Cisco Vlahakis
Wednesday (8-9 PM) @ Huang Basement
Or by appointment

Tiffany Manuel
Thursday (2-3 PM) @ Huang Basement
Or by appointment

For today's attendance, please see #general channel in our Slack.

Invitation:
https://tinyurl.com/cs47slack2019

CS47SI: Cross-Platform Mobile Development

Lecture 7B: Networking

James Landay
Abdallah AbuHashem
Tiffany Manuel
Cisco Vlahakis
Vy Mai

https://cs47.stanford.edu

cs47-fall19.slack.com

Fall 2019