Introduction to Context API

In this lesson, we'll look at the details of Context API.

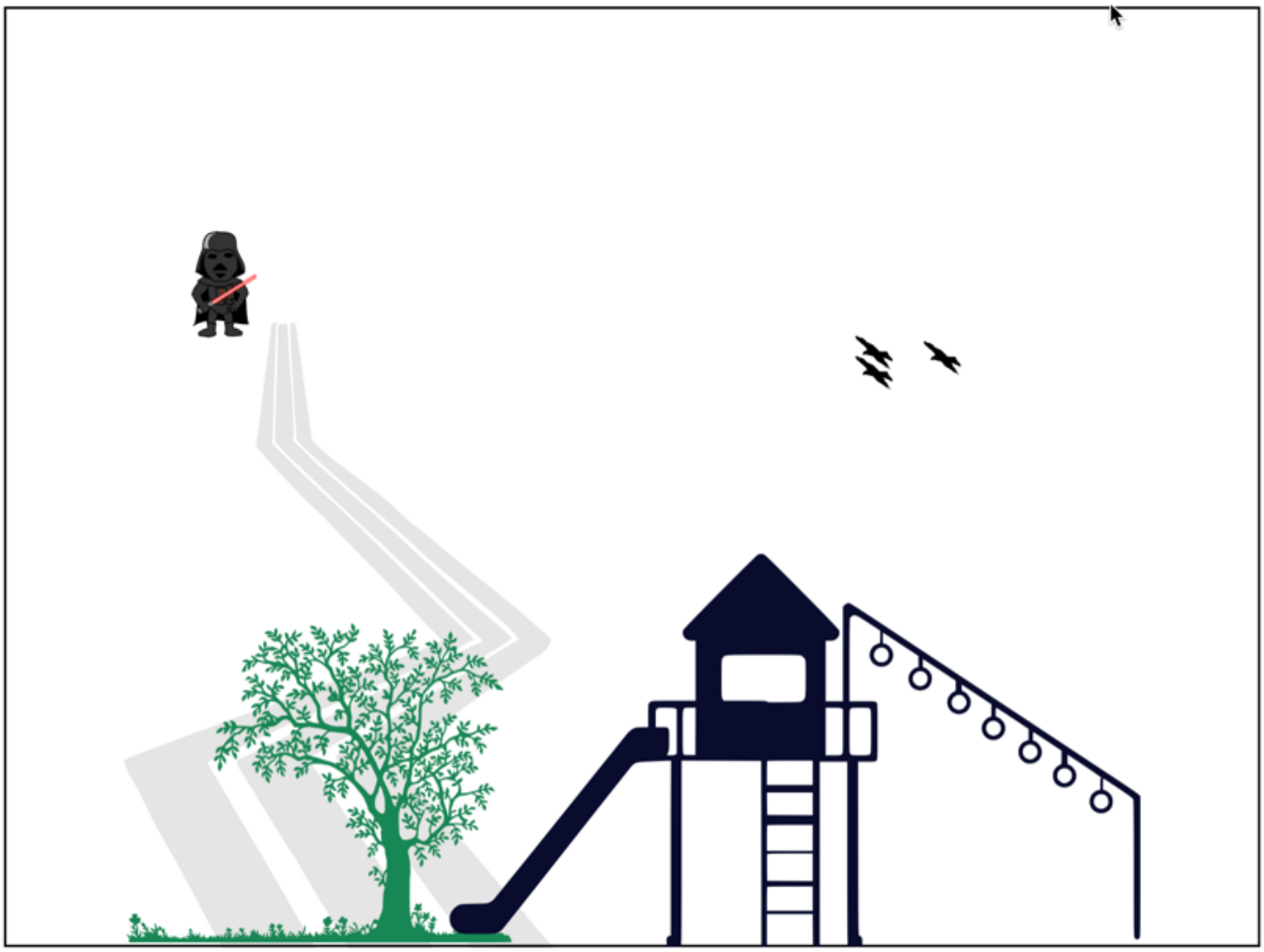
WE'LL COVER THE FOLLOWING

The Context API exists to make it easy to share data considered “global” within a component tree.

Let’s have a look at an illustrated example before we delve into writing any code.

John has begun working with the Context API and has been impressed with it so far. Today, he has a new project to work on, and he intends to use the context API.

Let’s see what this new project is about.



The new project: Benny Home Run

Benny Home Run Game

John is expected to build a game for a new client of his. This game is called **Benny Home Run**, and it seems like a great place to use the Context API.

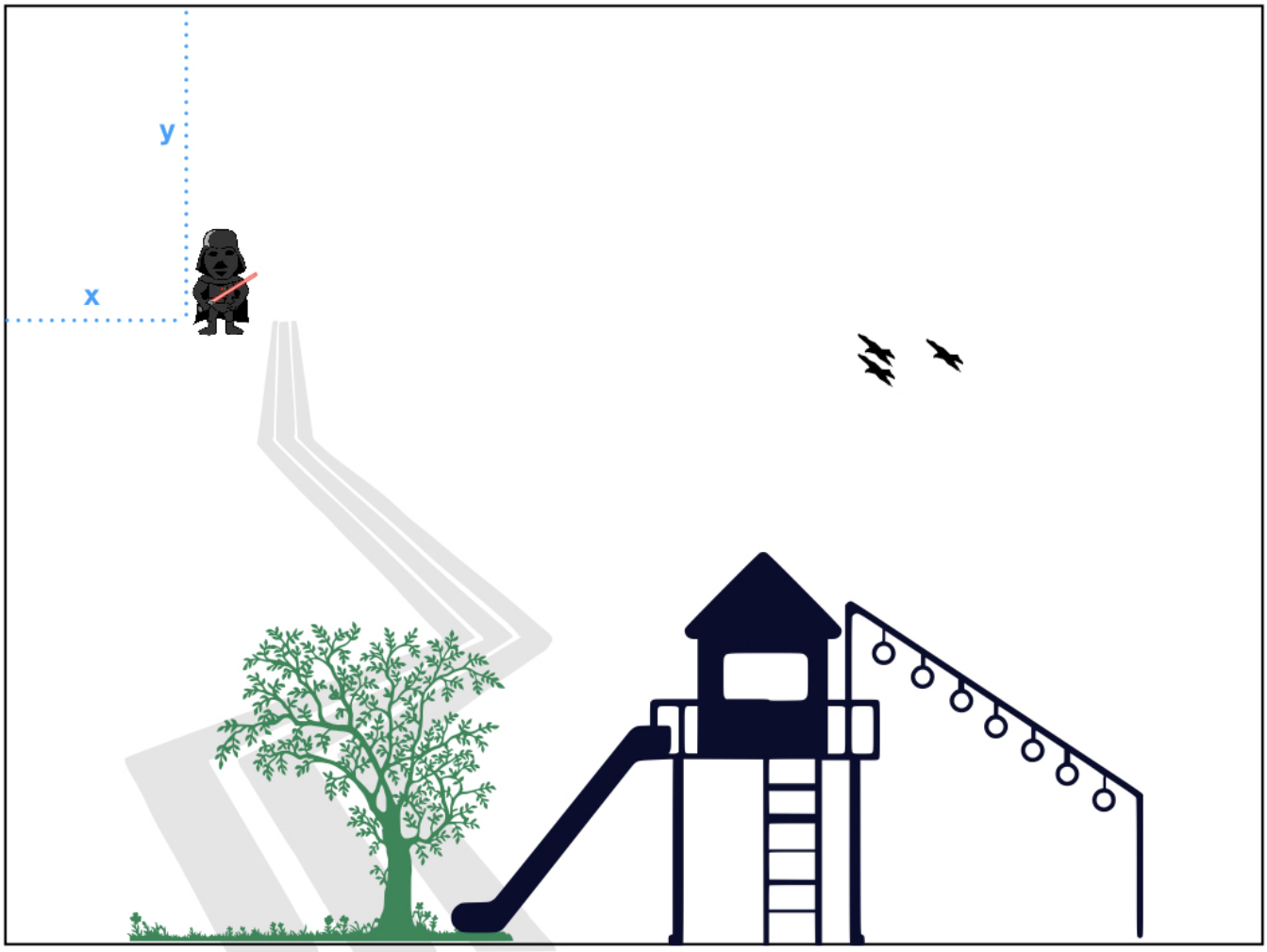
The aim of the game is to move Benny from his start position to his new home.

The aim of the game

Logic of the Game

To build this game, John must keep track of the Benny's position.

Since Benny's position is such an integral part of the entire application, it may as well be tracked via some global application state.



Tracking position values x and y in state

Did I just say *global application state*?

Yeah!

That sounds like a good fit for the Context API.

So, how could John build this?

```
import {createContext} from 'react';
const BennyPositionContext = createContext({ x: 50, y: 50 })
const { Provider, Consumer } = BennyPositionContext
```

First, he needs to import the `createContext` method from `React`

```
import {createContext} from 'react';
```

The `createContext` method allows you to create what's referred to as a context object. Consider this to be the data structure that powers retrieving and saving state values.

To create a context object, you invoke the `createContext` method with or without an initial state value to be saved in the context object.

```
createContext(initialStateValue)
```

Here's what that looks like in the Benny app:

```
const BennyPositionContext = createContext({
        x: 50,
        y: 50
})
```

The `createContext` method is invoked with an initial state value corresponding to the initial position values (`x` and `y`) for Benny.

Looks good!

But, after creating a context object, how exactly do you gain access to the state values within your application?