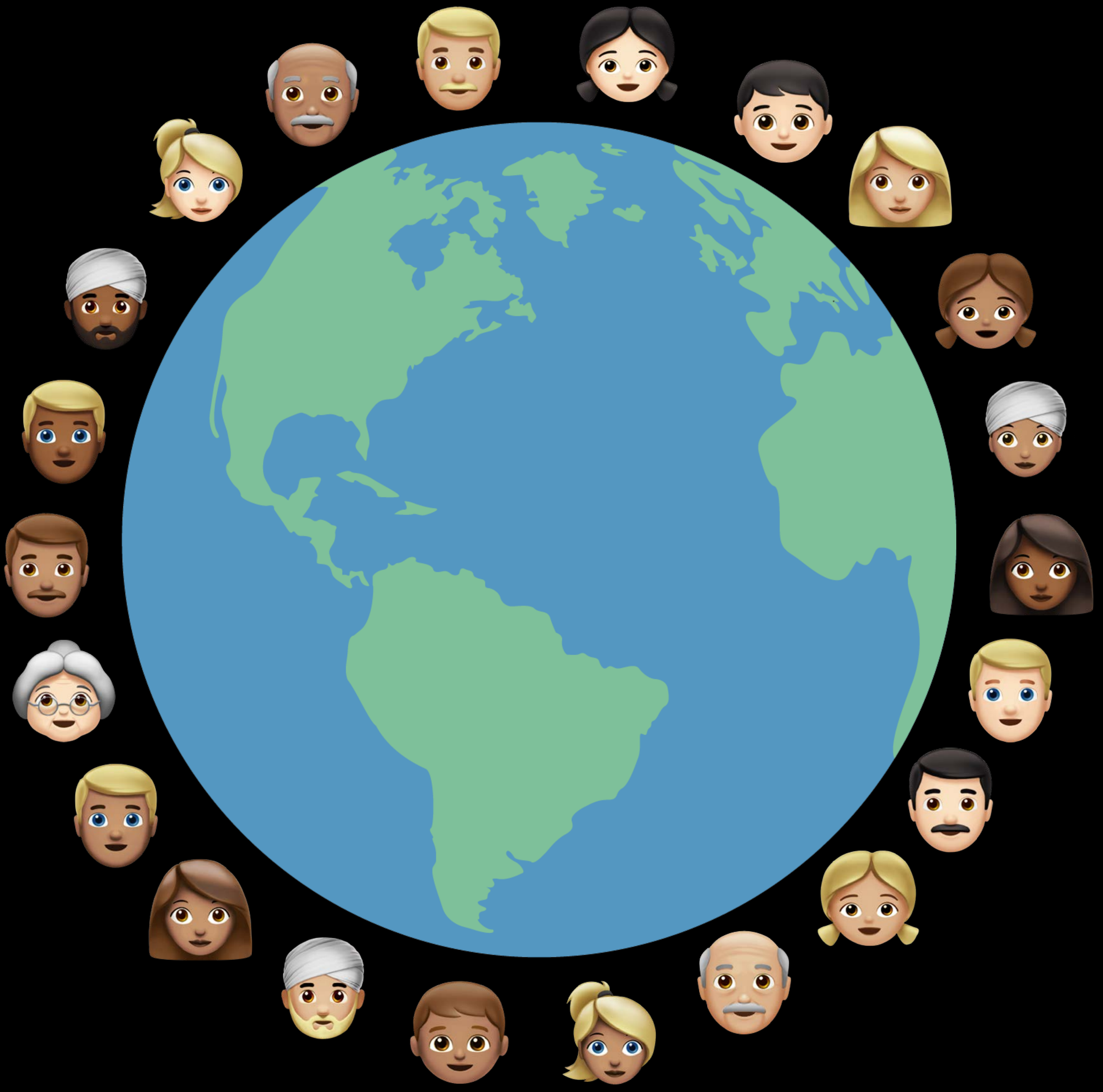
Developer Tools

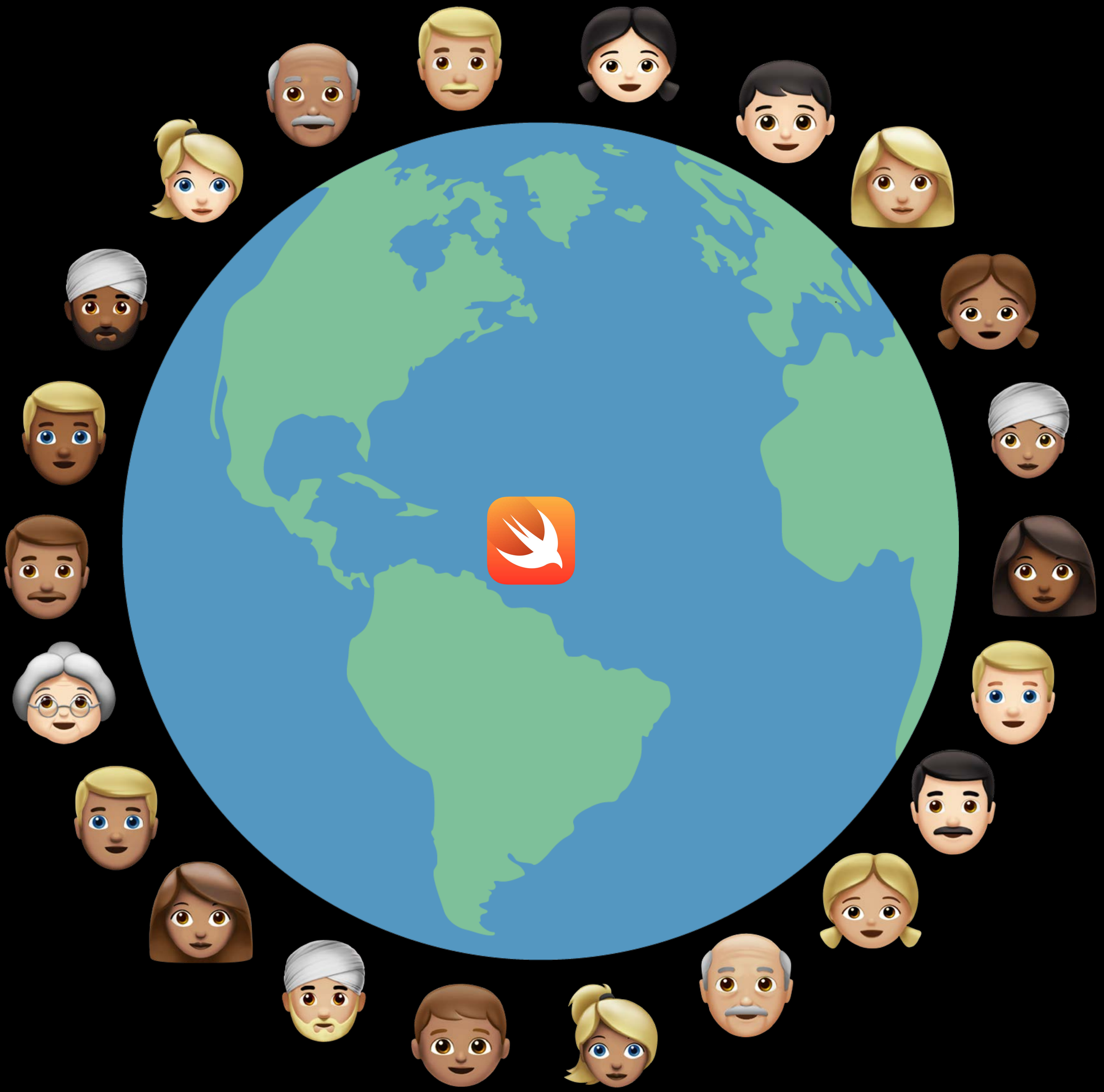
#WWDC17

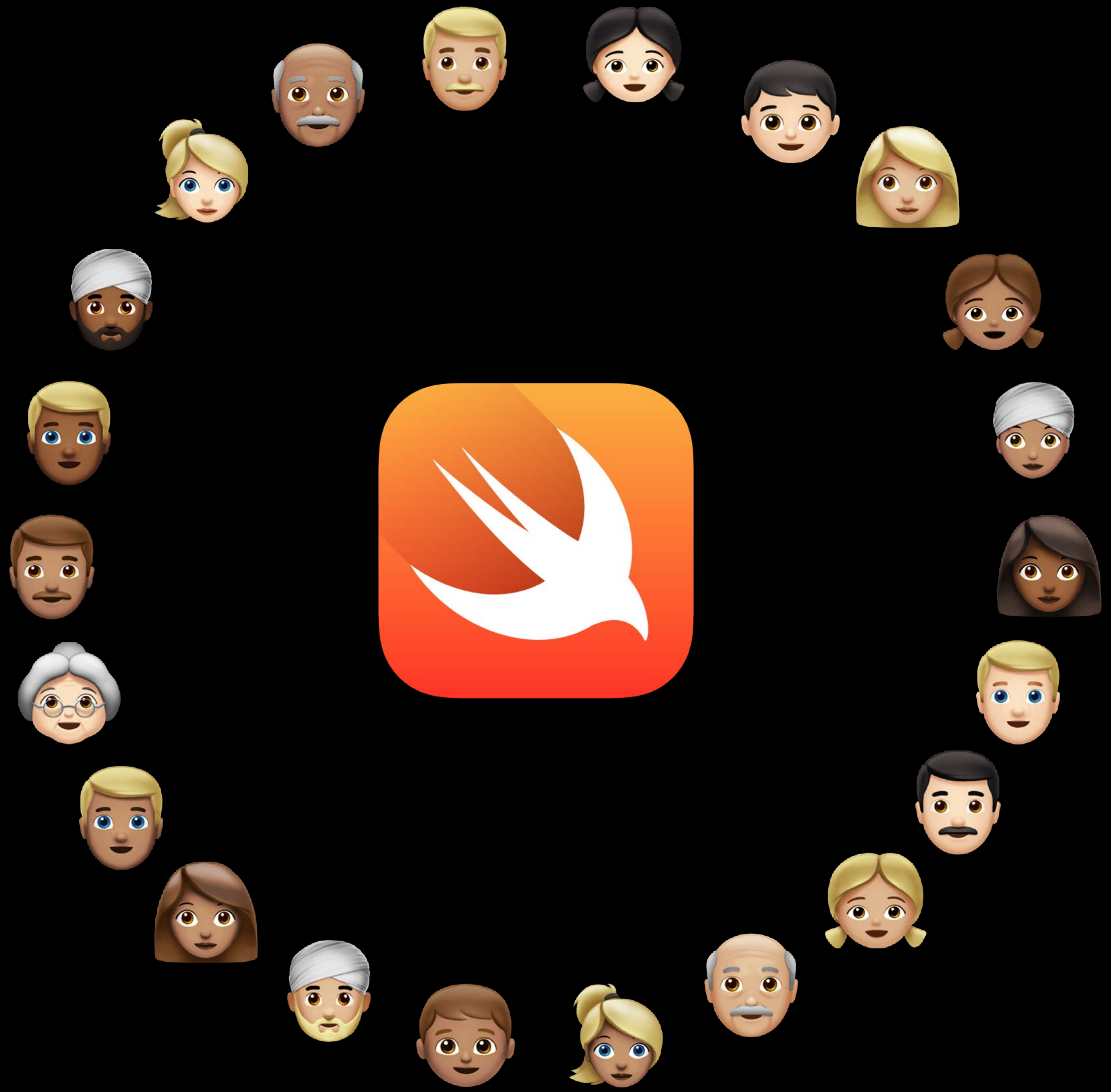
Localizing Content for Swift Playgrounds

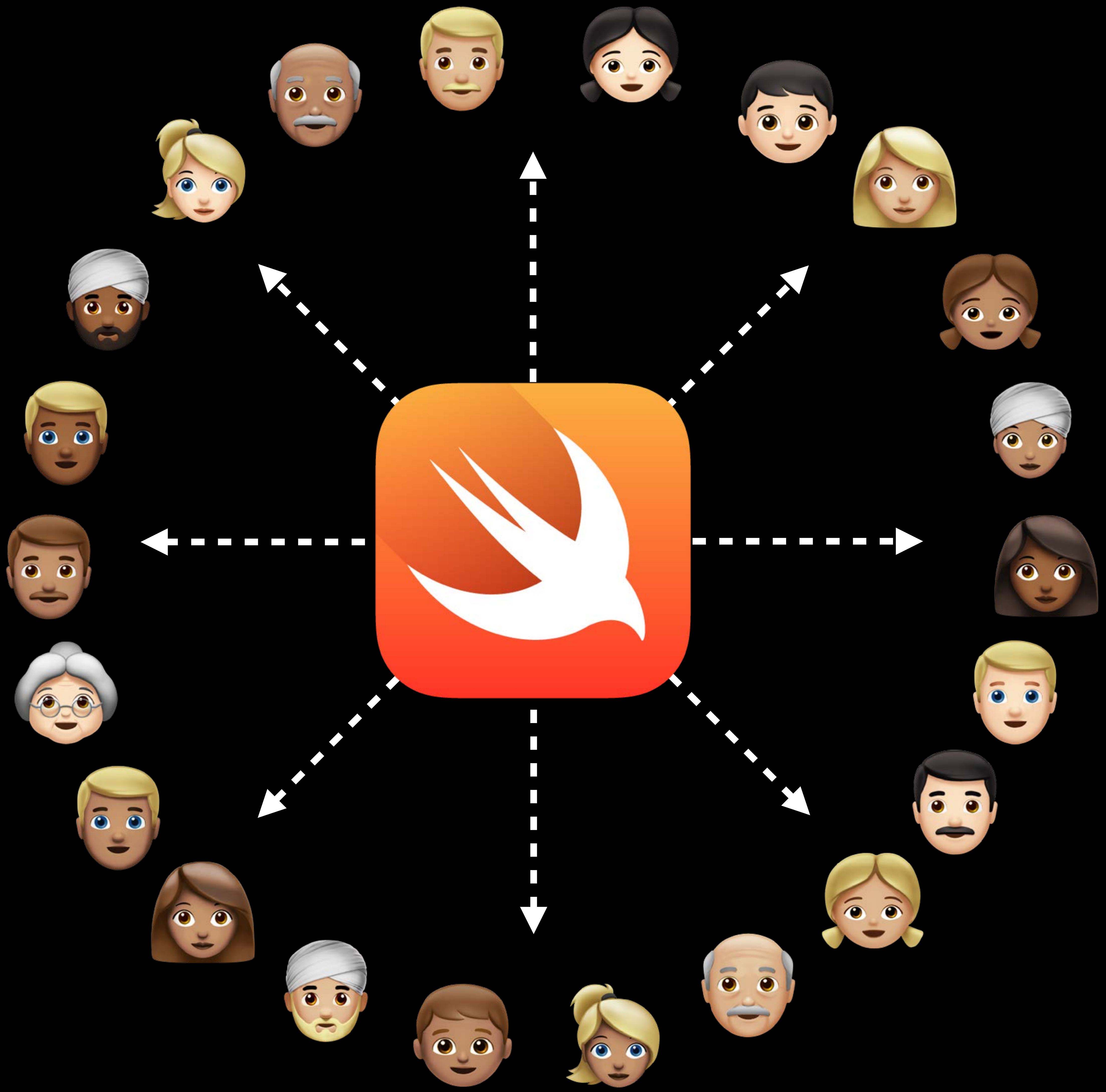
Session 410

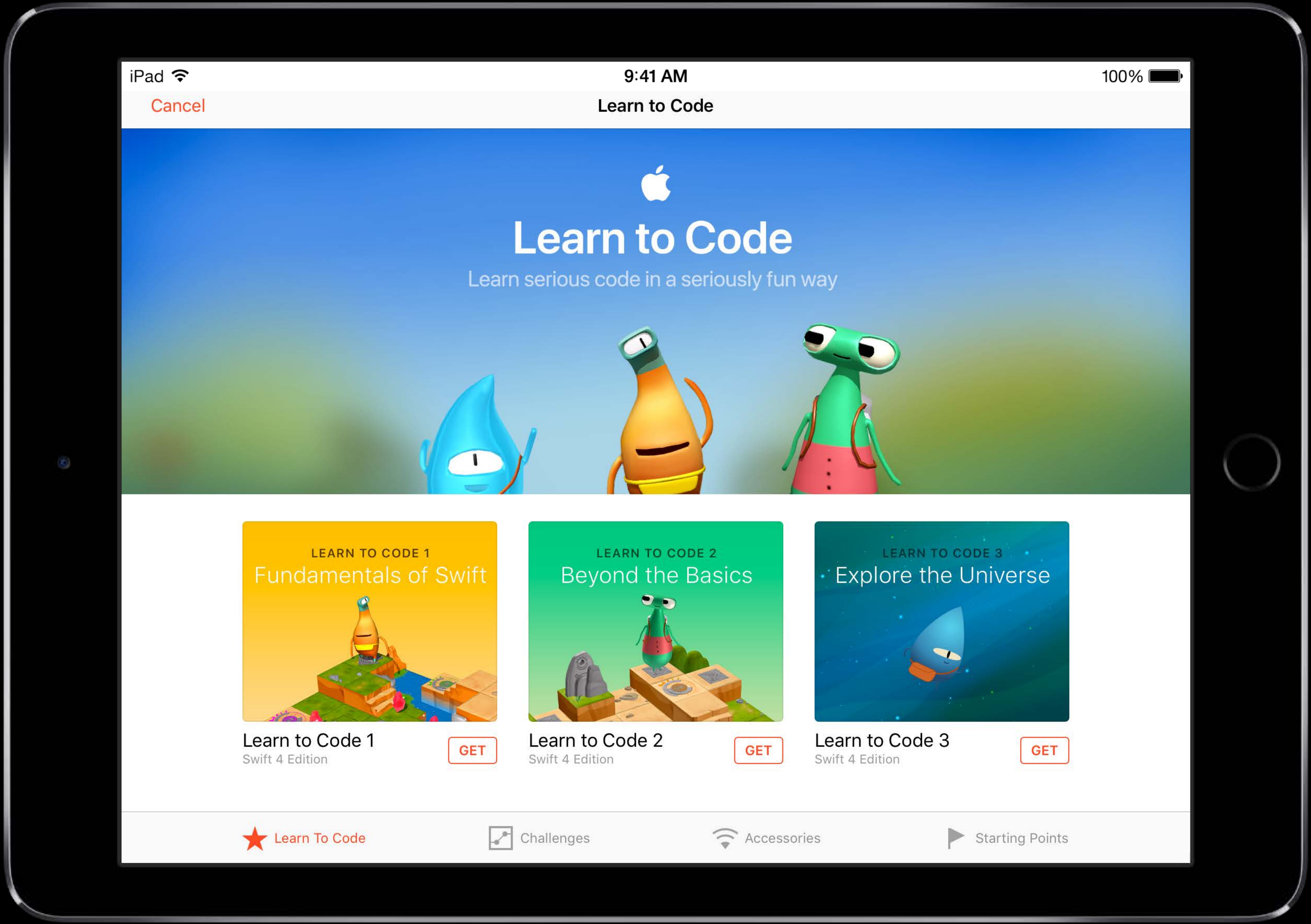
Alex Kuhn, Playgrounds Content Developer
Aaltan Ahmad, Internationalization Software Engineer











Learn to Code

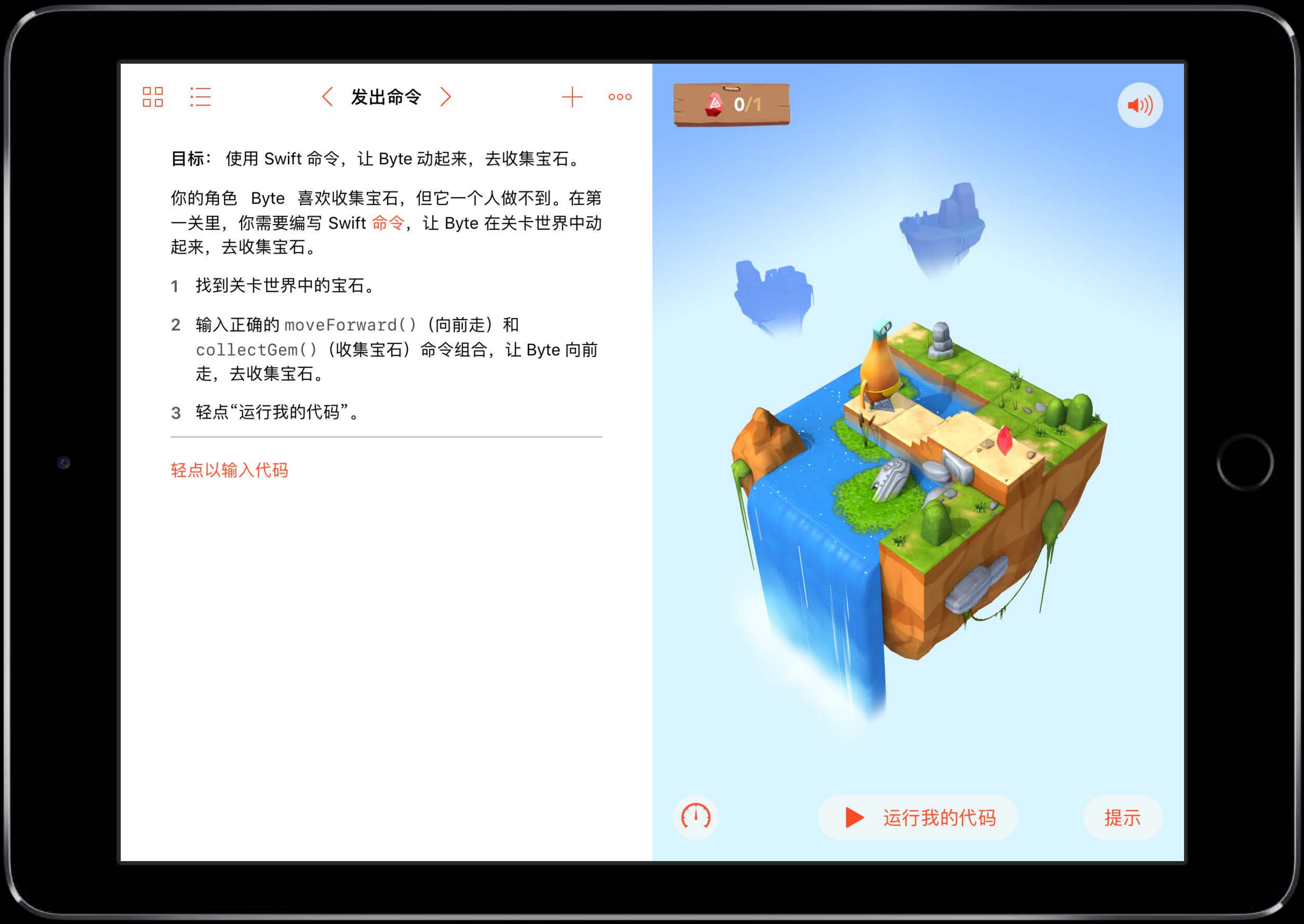
Learn serious code in a seriously fun way

LEARN TO CODE 1
Fundamentals of Swift

LEARN TO CODE 2
Beyond the Basics

LEARN TO CODE 3
Explore the Universe

Learn to Code 1 Swift 4 Edition	GET
Learn to Code 2 Swift 4 Edition	GET
Learn to Code 3 Swift 4 Edition	GET



发出命令

目标：使用 Swift 命令，让 Byte 动起来，去收集宝石。

你的角色 Byte 喜欢收集宝石，但它一个人做不到。在第一关里，你需要编写 Swift 命令，让 Byte 在关卡世界中动起来，去收集宝石。

1 找到关卡世界中的宝石。

2 输入正确的 `moveForward()`（向前走）和 `collectGem()`（收集宝石）命令组合，让 Byte 向前走，去收集宝石。

3 轻点“运行我的代码”。

轻点以输入代码

Agenda

Agenda

Overview

Agenda

Overview

Localization considerations

Agenda

Overview

Localization considerations

Structure of a localized playground book

Define your content goals

Learn to Code Goals

Learn to Code Goals

Instructional

Learn to Code Goals

Instructional

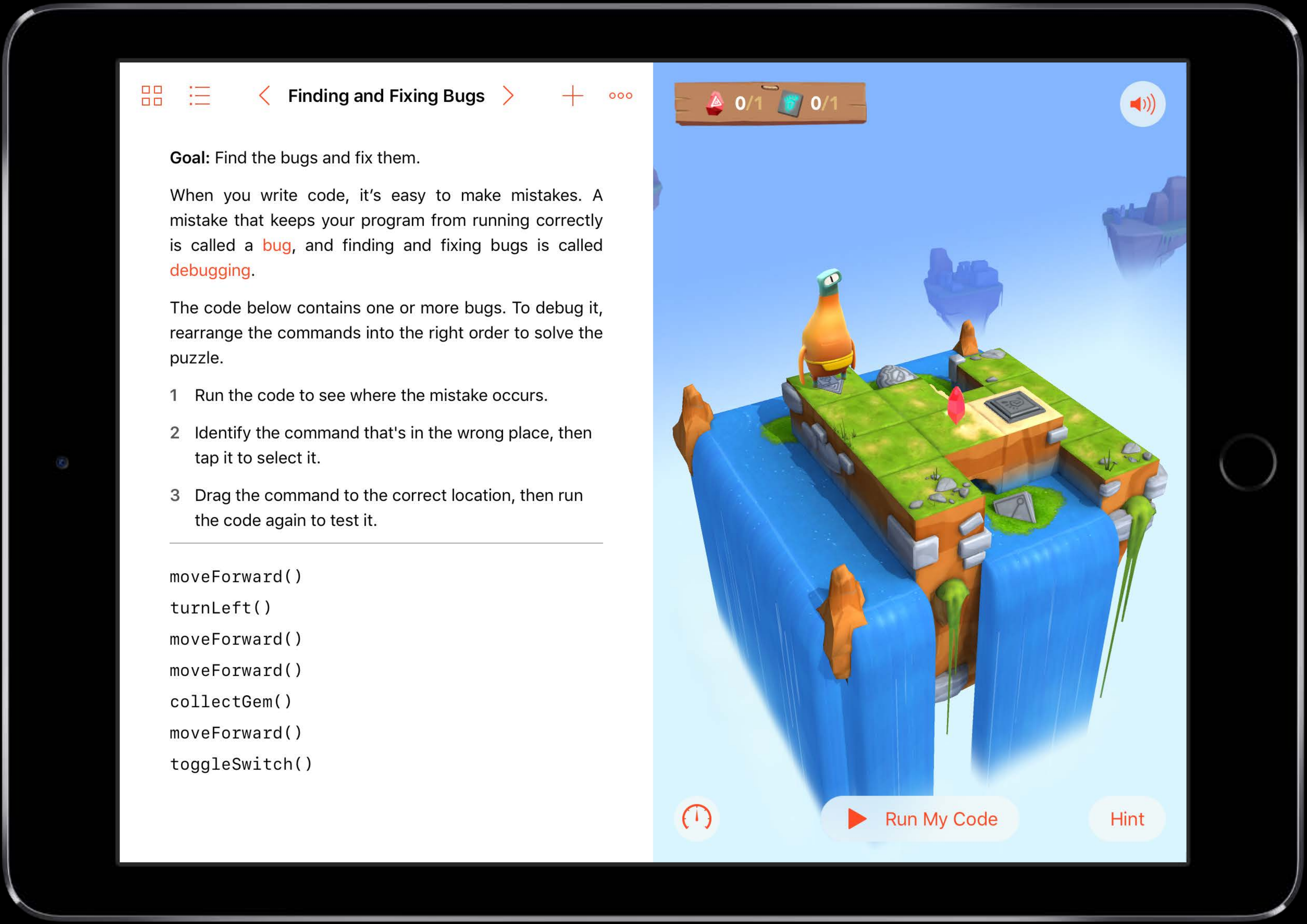
Fun and engaging

Learn to Code Goals

Instructional

Fun and engaging

Ages 12 and up



Finding and Fixing Bugs

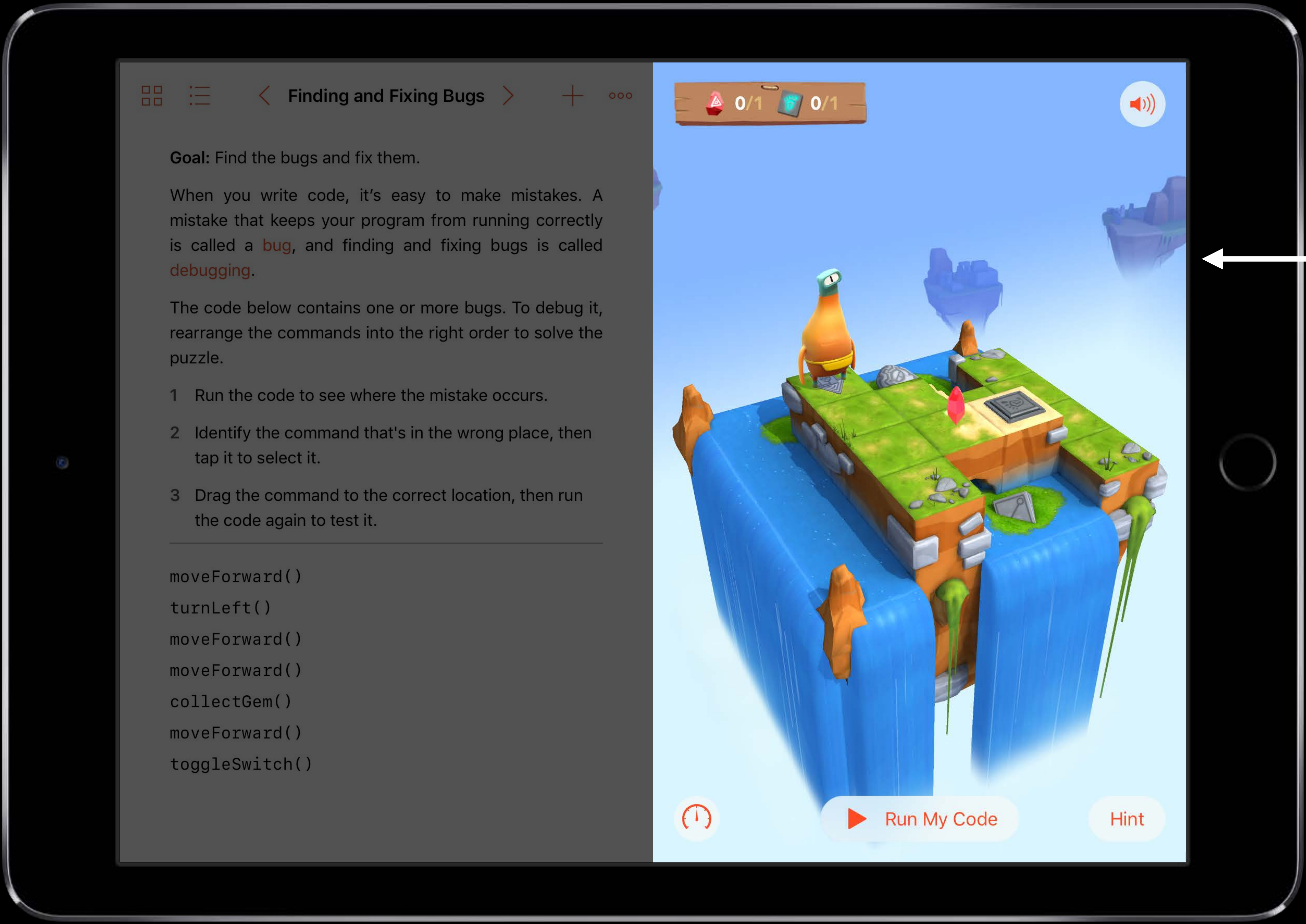
Goal: Find the bugs and fix them.

When you write code, it’s easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

1. Run the code to see where the mistake occurs.
2. Identify the command that's in the wrong place, then tap it to select it.
3. Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```



Goal: Find the bugs and fix them.

When you write code, it's easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

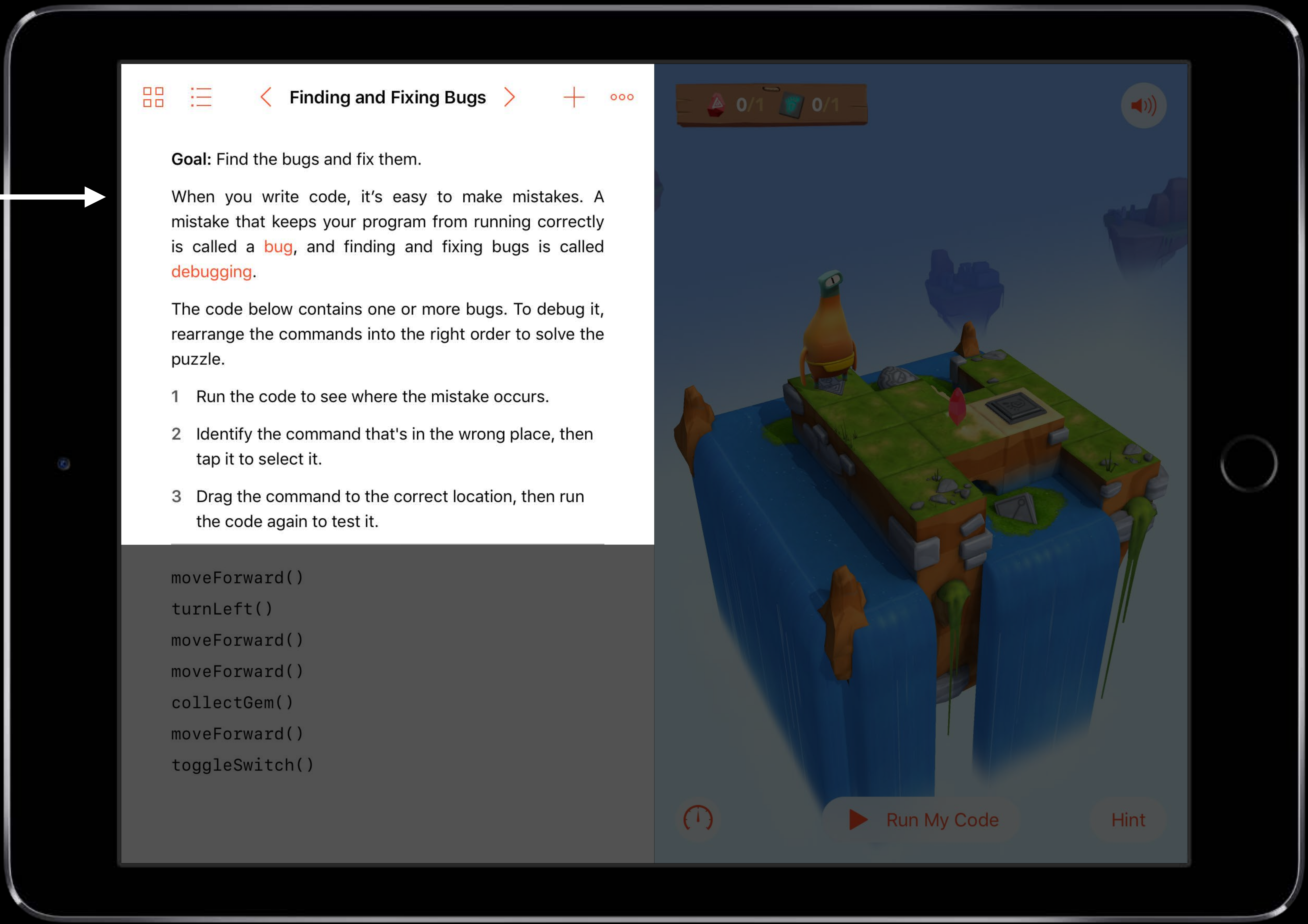
The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

1. Run the code to see where the mistake occurs.
2. Identify the command that's in the wrong place, then tap it to select it.
3. Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

Live View

Prose



Goal: Find the bugs and fix them.

When you write code, it’s easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

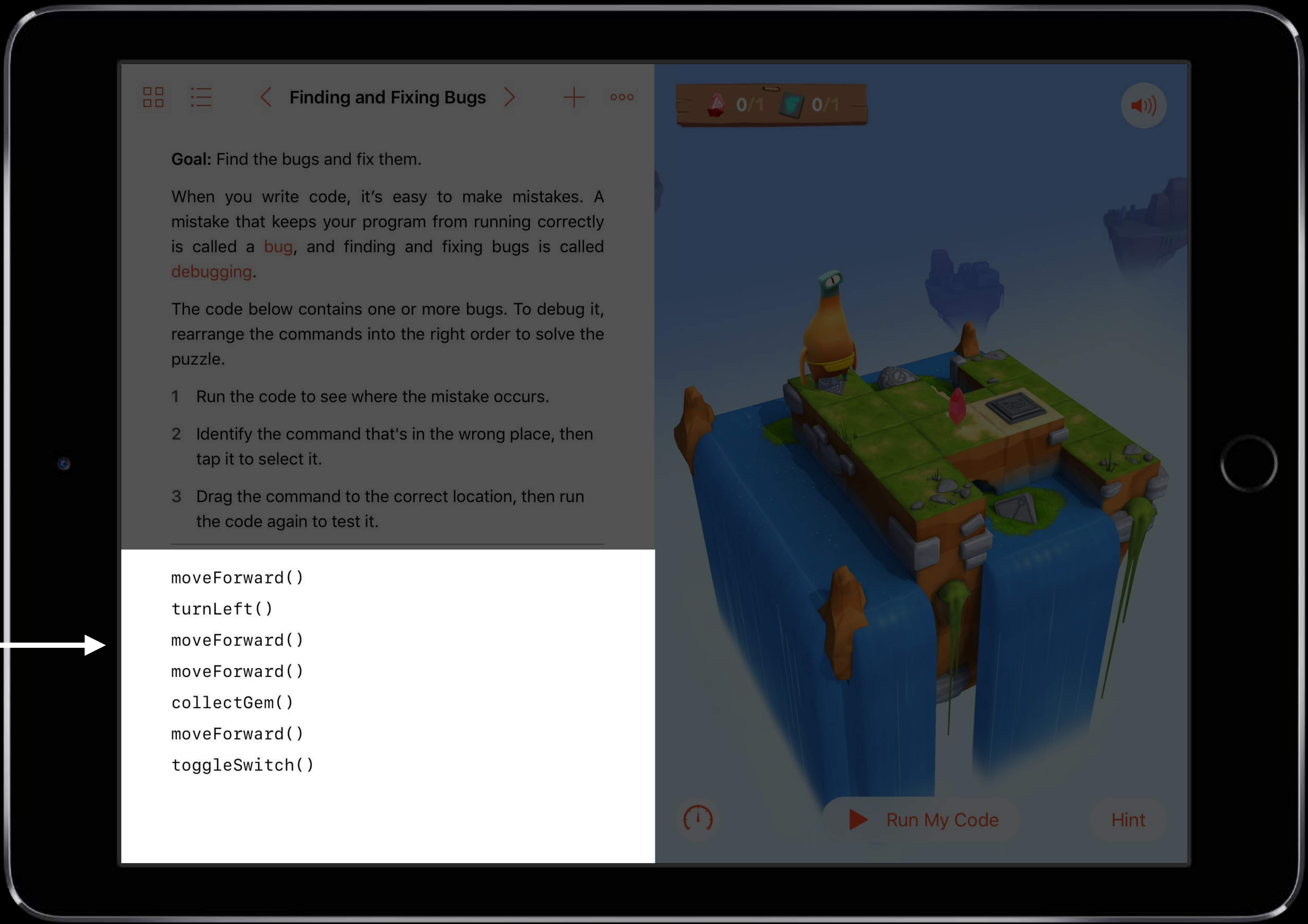
The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

1 Run the code to see where the mistake occurs.

2 Identify the command that's in the wrong place, then tap it to select it.

3 Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```



Goal: Find the bugs and fix them.

When you write code, it's easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

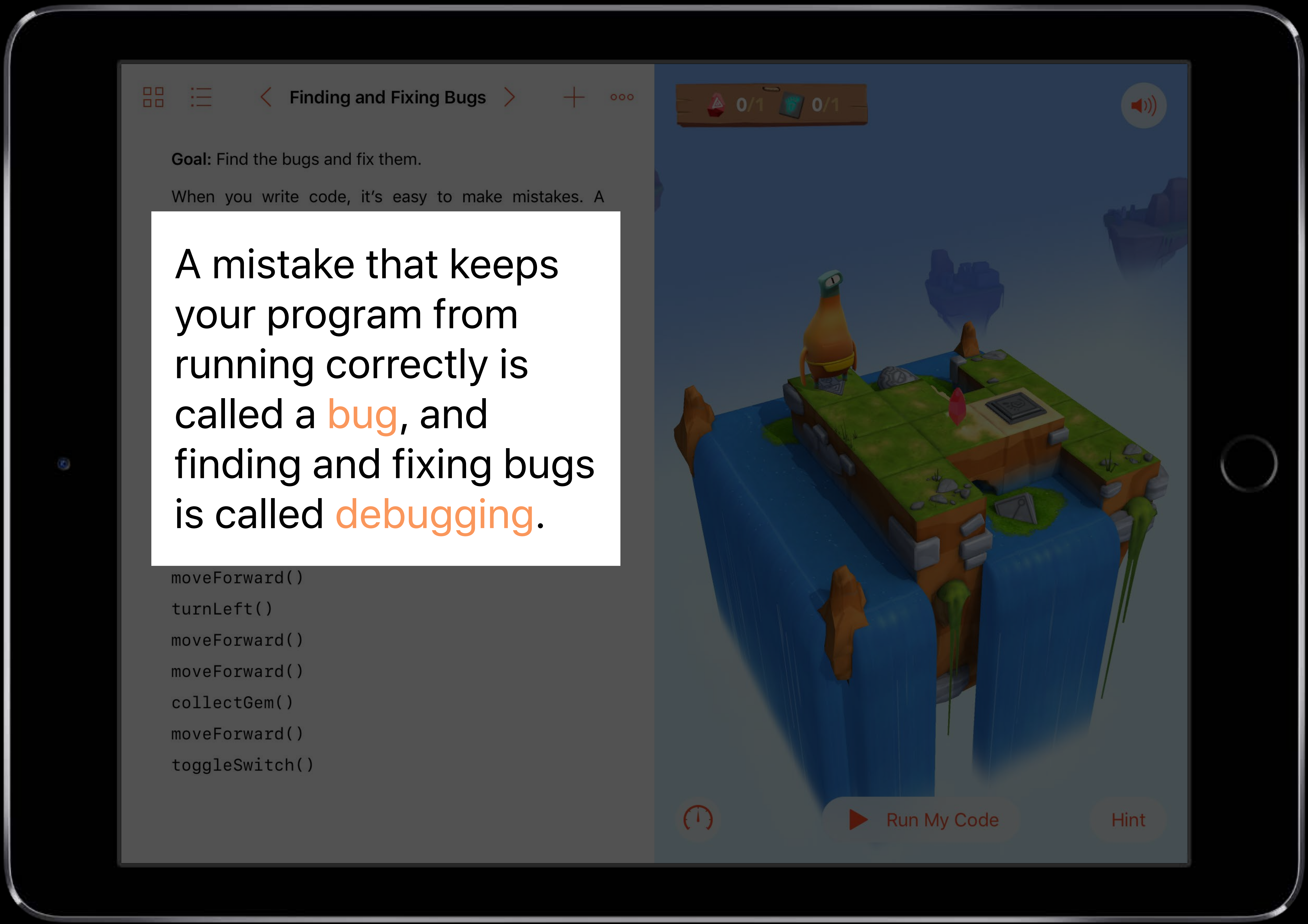
1 Run the code to see where the mistake occurs.

2 Identify the command that's in the wrong place, then tap it to select it.

3 Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

Code

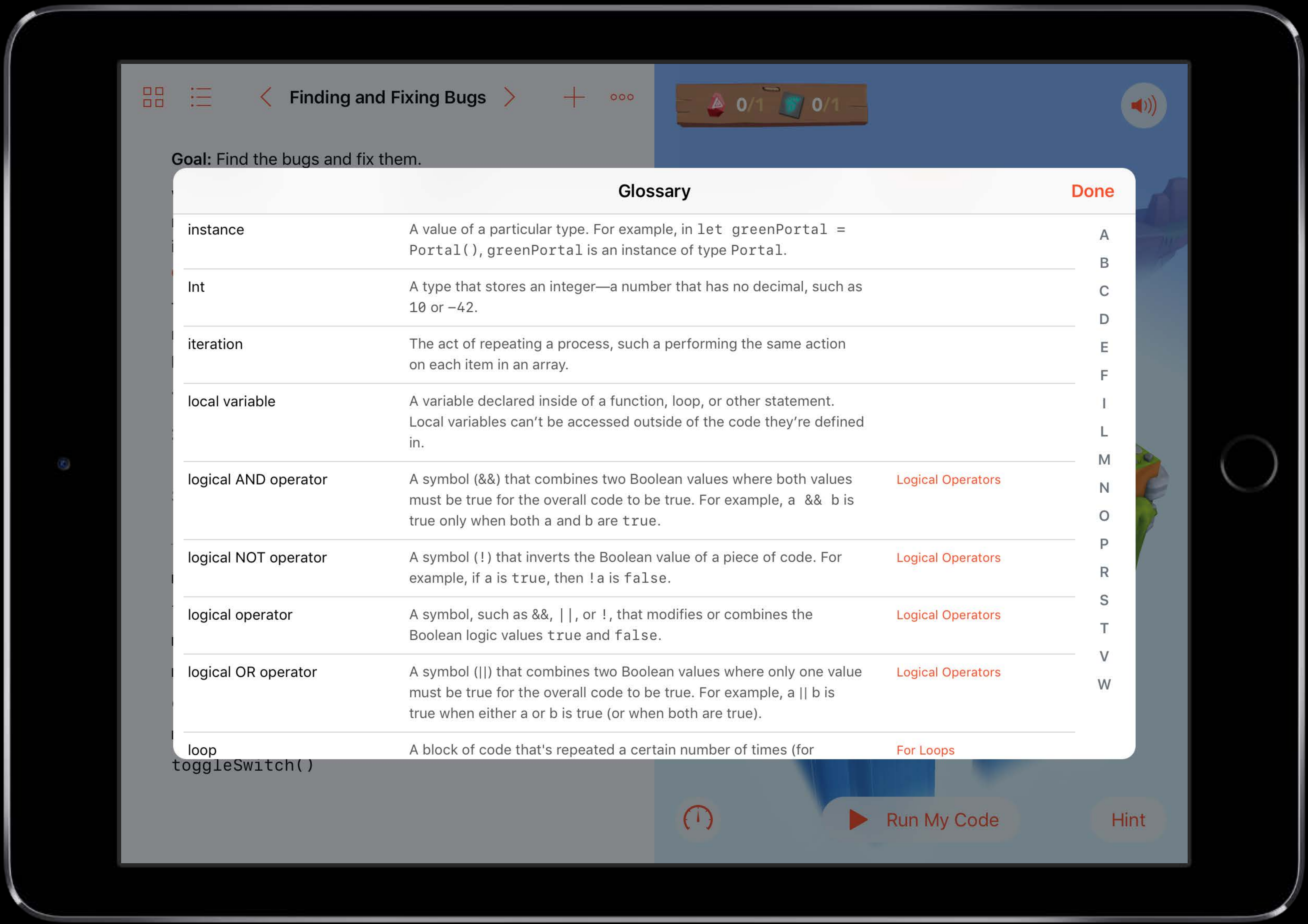


Goal: Find the bugs and fix them.

When you write code, it's easy to make mistakes. A

A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```



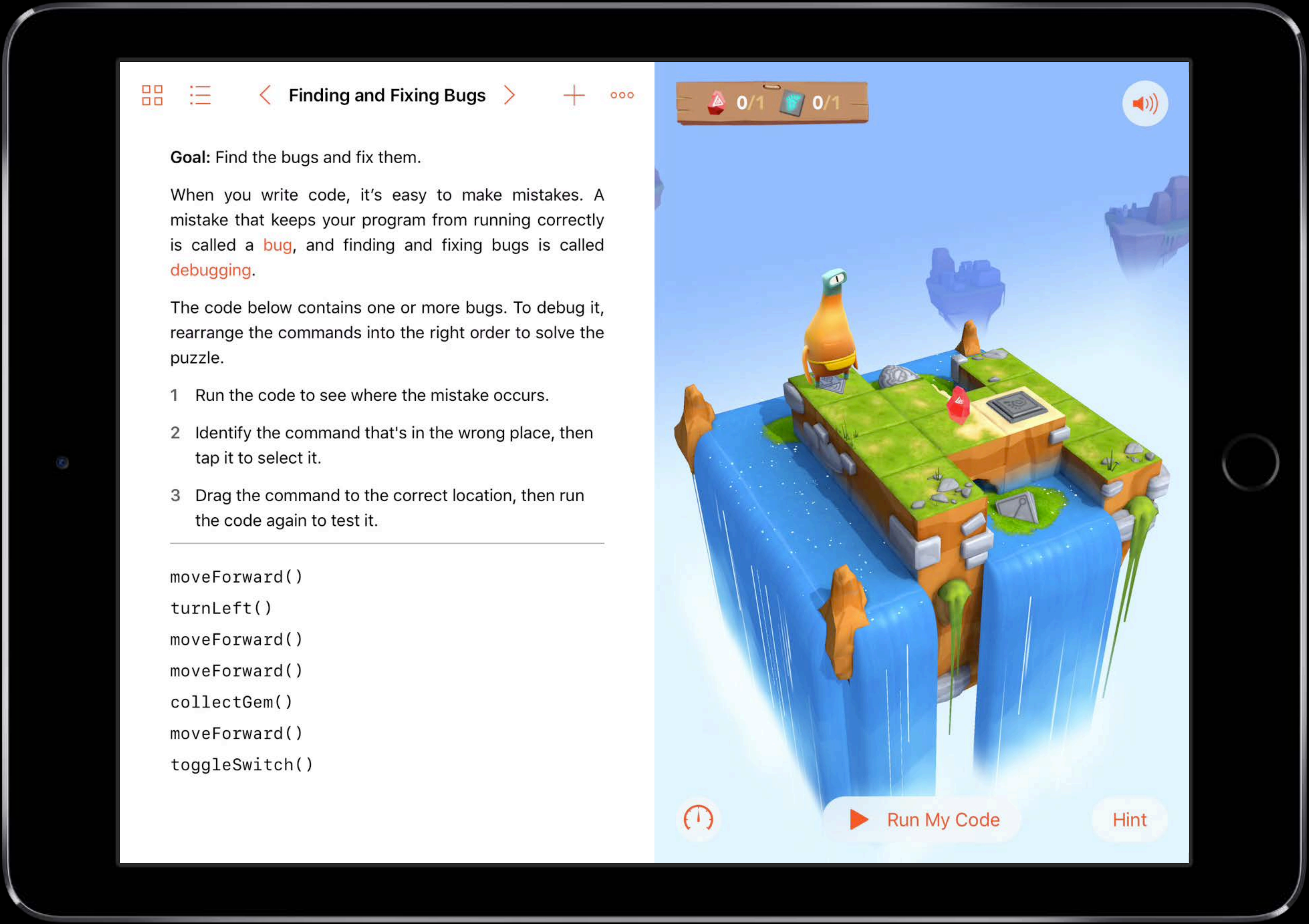
Goal: Find the bugs and fix them.

Glossary

Done

instance	A value of a particular type. For example, in `let greenPortal = Portal()`, `greenPortal` is an instance of type `Portal`.	
Int	A type that stores an integer—a number that has no decimal, such as `10` or `−42`.	
iteration	The act of repeating a process, such a performing the same action on each item in an array.	
local variable	A variable declared inside of a function, loop, or other statement. Local variables can't be accessed outside of the code they're defined in.	
logical AND operator	A symbol (&&) that combines two Boolean values where both values must be true for the overall code to be true. For example, a `&&` b is true only when both a and b are `true`.	Logical Operators
logical NOT operator	A symbol (`!`) that inverts the Boolean value of a piece of code. For example, if a is `true`, then `!`a is `false`.	Logical Operators
logical operator	A symbol, such as &&, `||`, or `!`, that modifies or combines the Boolean logic values `true` and `false`.	Logical Operators
logical OR operator	A symbol (||) that combines two Boolean values where only one value must be true for the overall code to be true. For example, a || b is true when either a or b is true (or when both are true).	Logical Operators
loop	A block of code that's repeated a certain number of times (for	For Loops

```
toggleSwitch()
```



Finding and Fixing Bugs

Goal: Find the bugs and fix them.

When you write code, it's easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

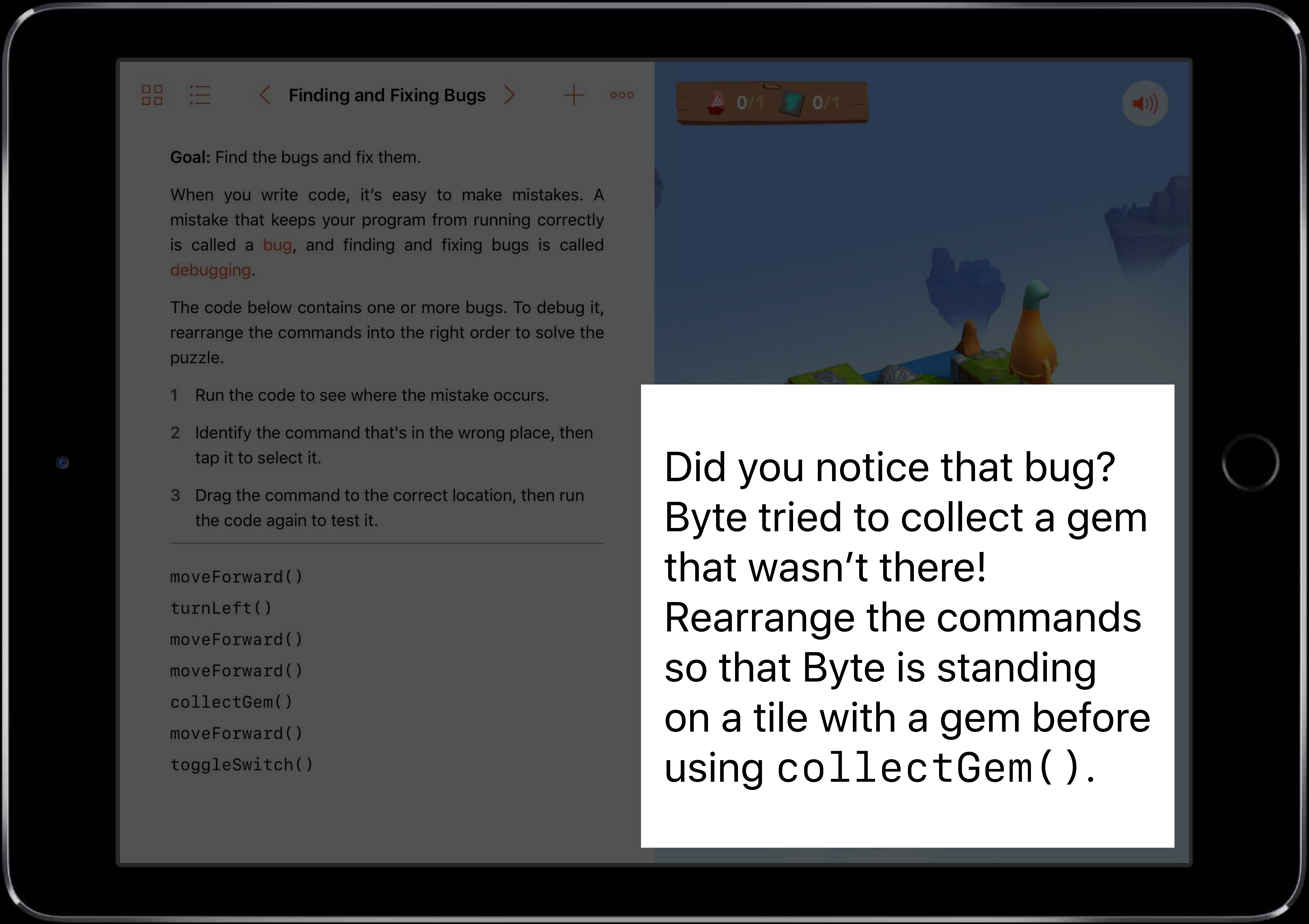
The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

1 Run the code to see where the mistake occurs.

2 Identify the command that's in the wrong place, then tap it to select it.

3 Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```



Goal: Find the bugs and fix them.

When you write code, it's easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

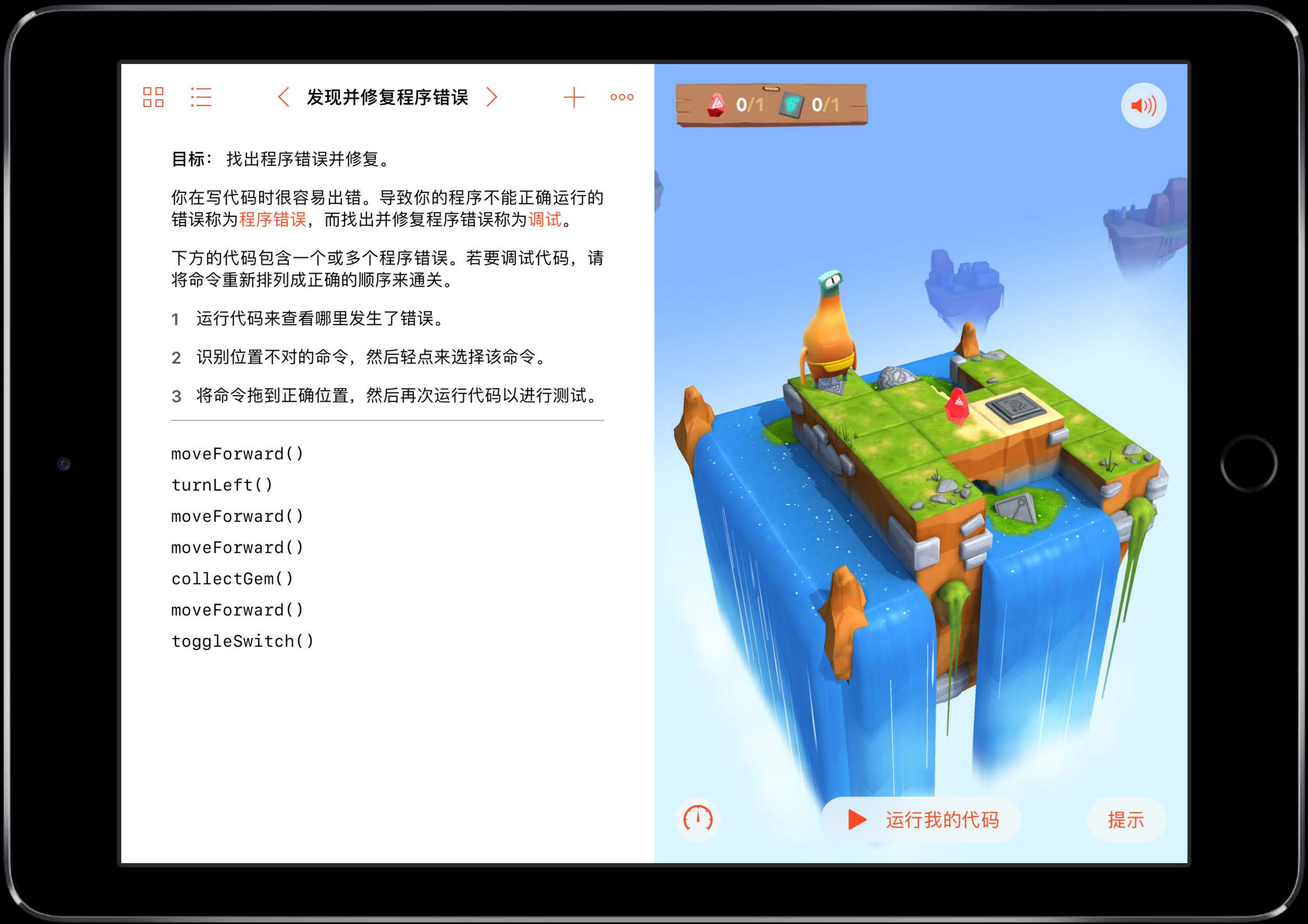
1 Run the code to see where the mistake occurs.

2 Identify the command that's in the wrong place, then tap it to select it.

3 Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

Did you notice that bug? Byte tried to collect a gem that wasn't there! Rearrange the commands so that Byte is standing on a tile with a gem before using `collectGem()`.



发现并修复程序错误

目标： 找出程序错误并修复。

你在写代码时很容易出错。导致你的程序不能正确运行的错误称为程序错误，而找出并修复程序错误称为调试。

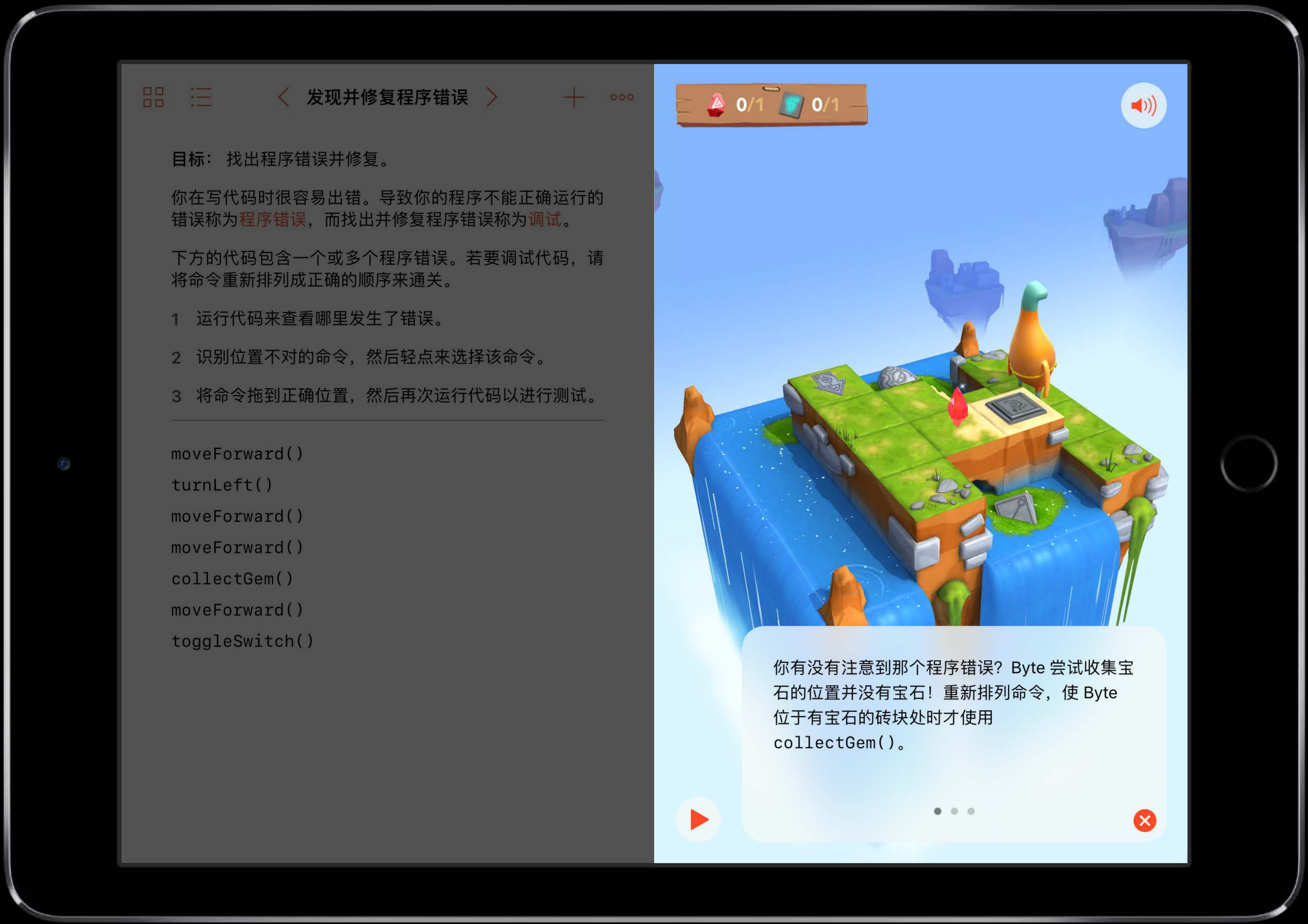
下方的代码包含一个或多个程序错误。若要调试代码，请将命令重新排列成正确的顺序来通关。

1 运行代码来查看哪里发生了错误。

2 识别位置不对的命令，然后轻点来选择该命令。

3 将命令拖到正确位置，然后再次运行代码以进行测试。

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```



发现并修复程序错误

目标：找出程序错误并修复。

你在写代码时很容易出错。导致你的程序不能正确运行的错误称为程序错误，而找出并修复程序错误称为调试。

下方的代码包含一个或多个程序错误。若要调试代码，请将命令重新排列成正确的顺序来通关。

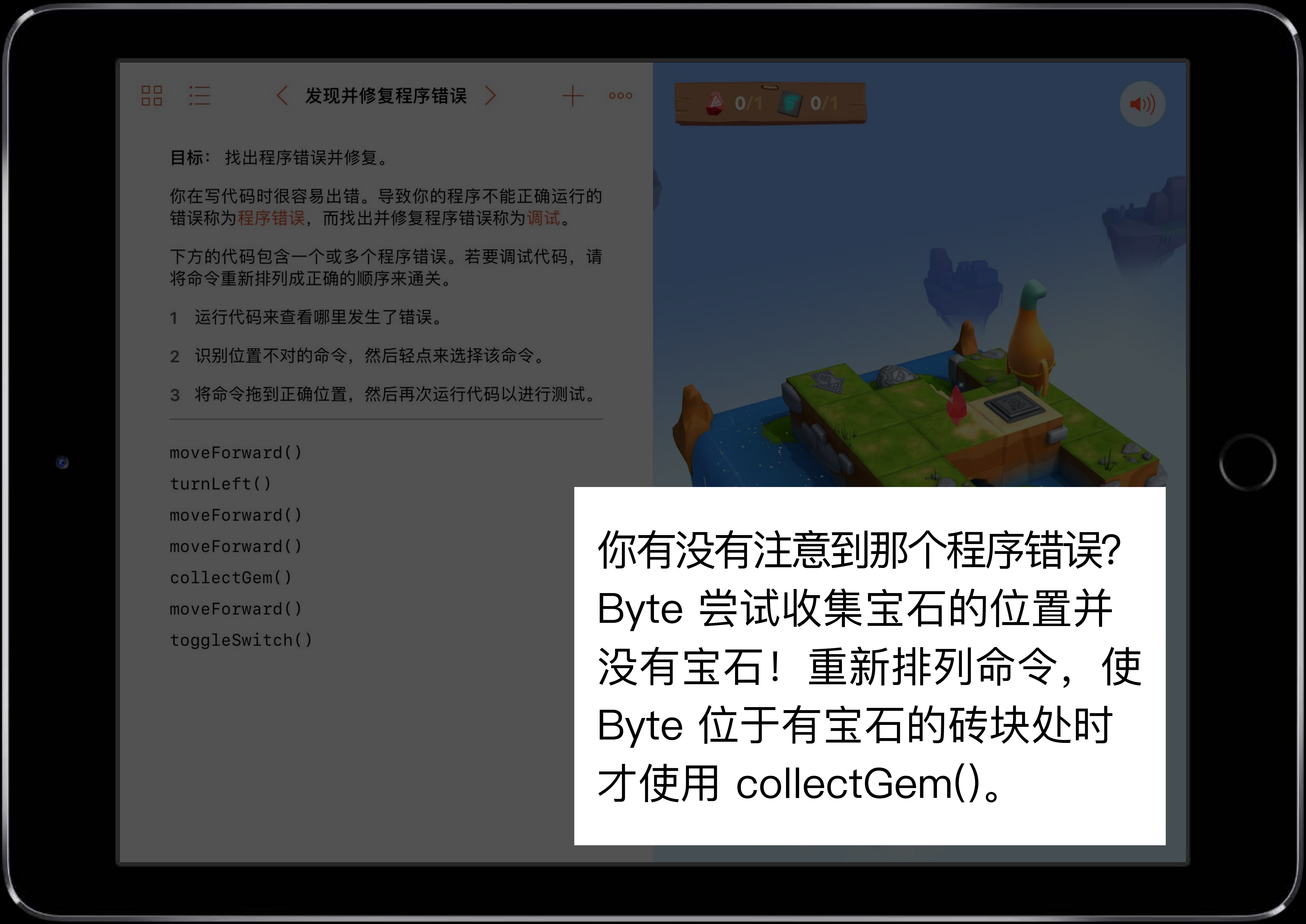
1 运行代码来查看哪里发生了错误。

2 识别位置不对的命令，然后轻点来选择该命令。

3 将命令拖到正确位置，然后再次运行代码以进行测试。

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

你有没有注意到那个程序错误？Byte 尝试收集宝石的位置并没有宝石！重新排列命令，使 Byte 位于有宝石的砖块处时才使用 collectGem()。



目标：找出程序错误并修复。

你在写代码时很容易出错。导致你的程序不能正确运行的错误称为程序错误，而找出并修复程序错误称为调试。

下方的代码包含一个或多个程序错误。若要调试代码，请将命令重新排列成正确的顺序来通关。

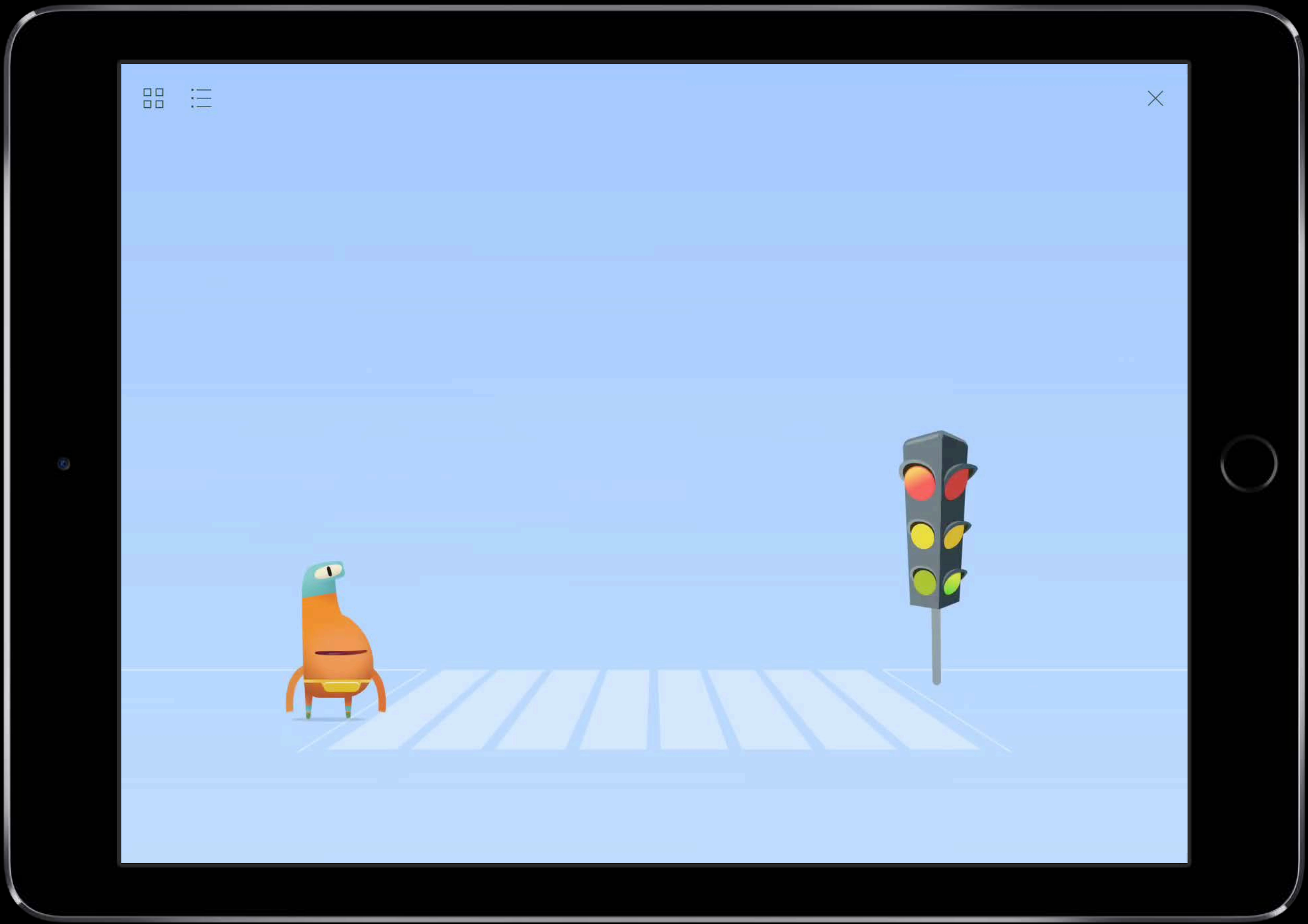
1 运行代码来查看哪里发生了错误。

2 识别位置不对的命令，然后轻点来选择该命令。

3 将命令拖到正确位置，然后再次运行代码以进行测试。

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

你有没有注意到那个程序错误? Byte 尝试收集宝石的位置并没有宝石！重新排列命令，使 Byte 位于有宝石的砖块处时才使用 collectGem()。



Building Localized Content

Building Localized Content

Goals and content

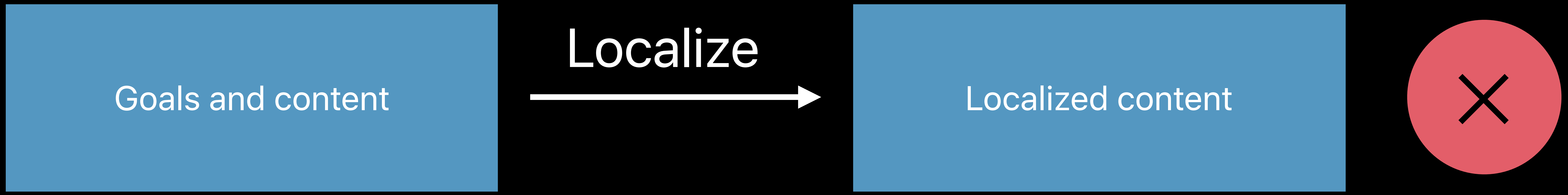
Building Localized Content

Goals and content

Localize

Localized content

Building Localized Content



Goals and content

Localize

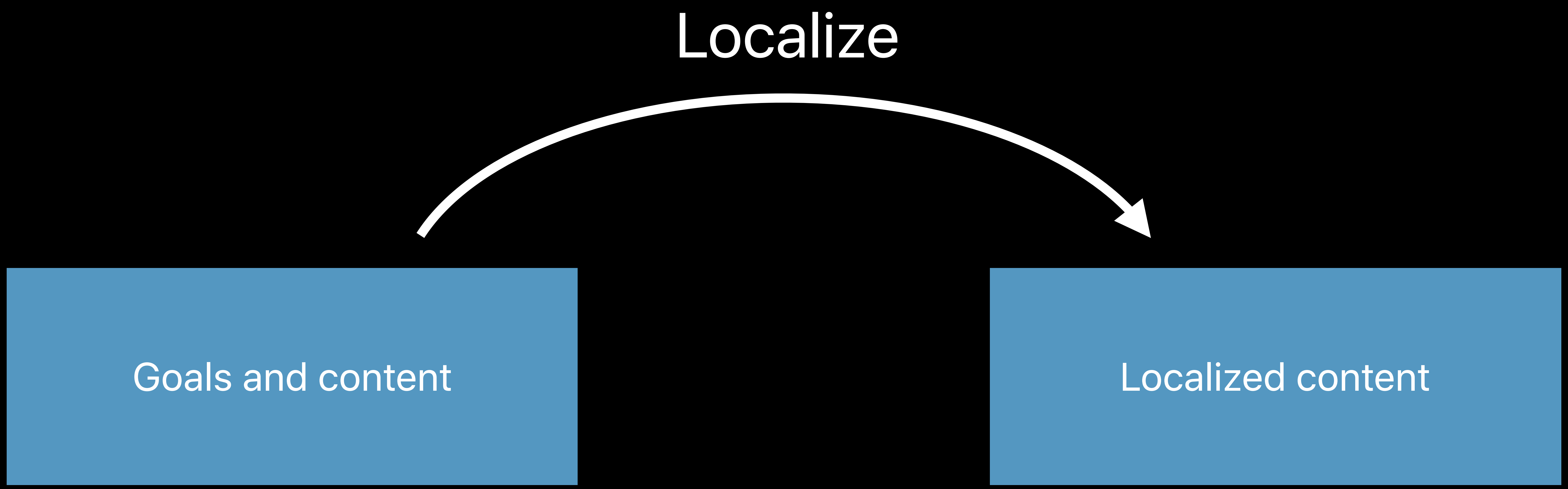
Localized content

Localization is part of design

Building Localized Content

Goals and content

Building Localized Content

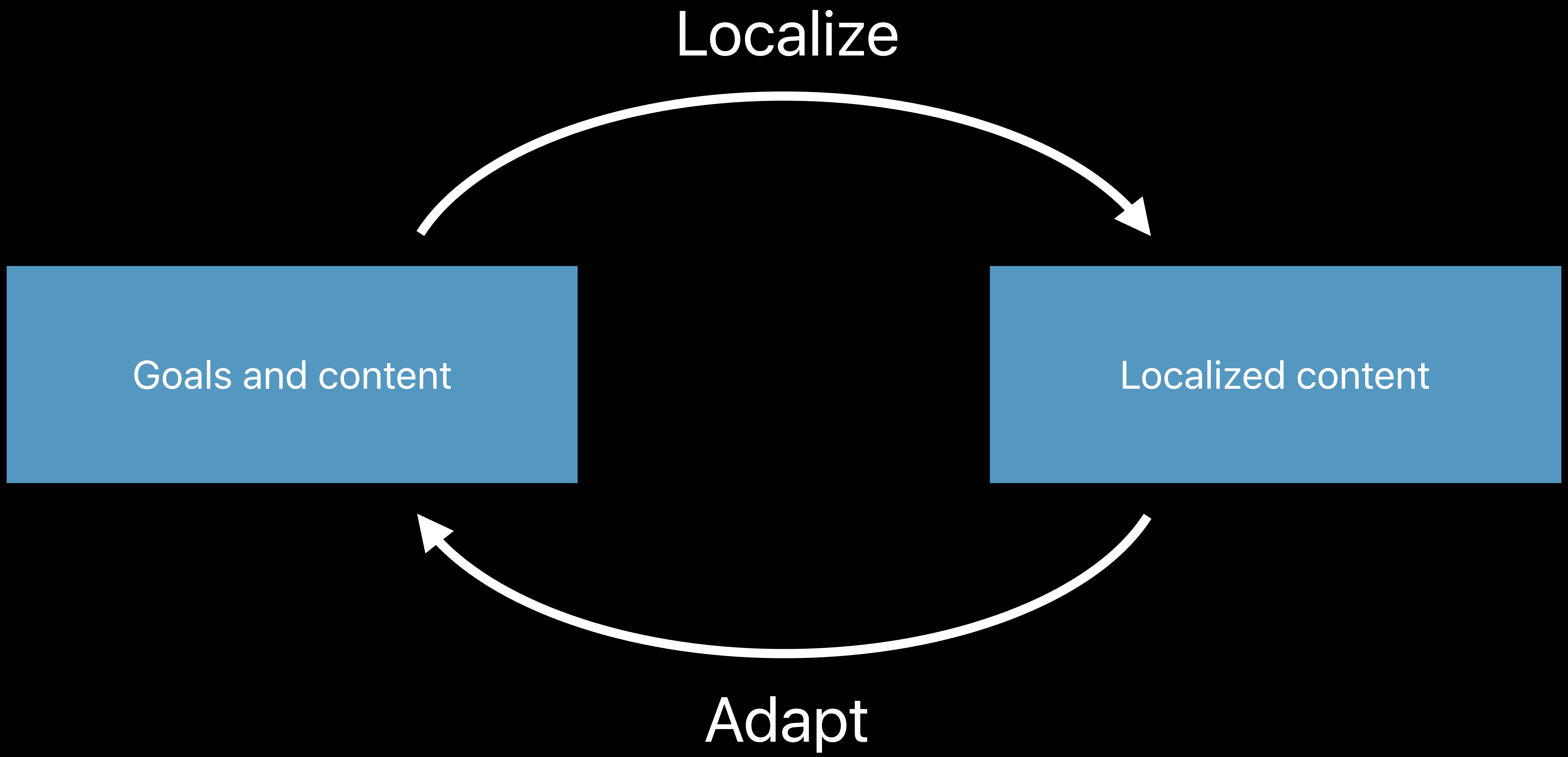


Localize

Goals and content

Localized content

Building Localized Content



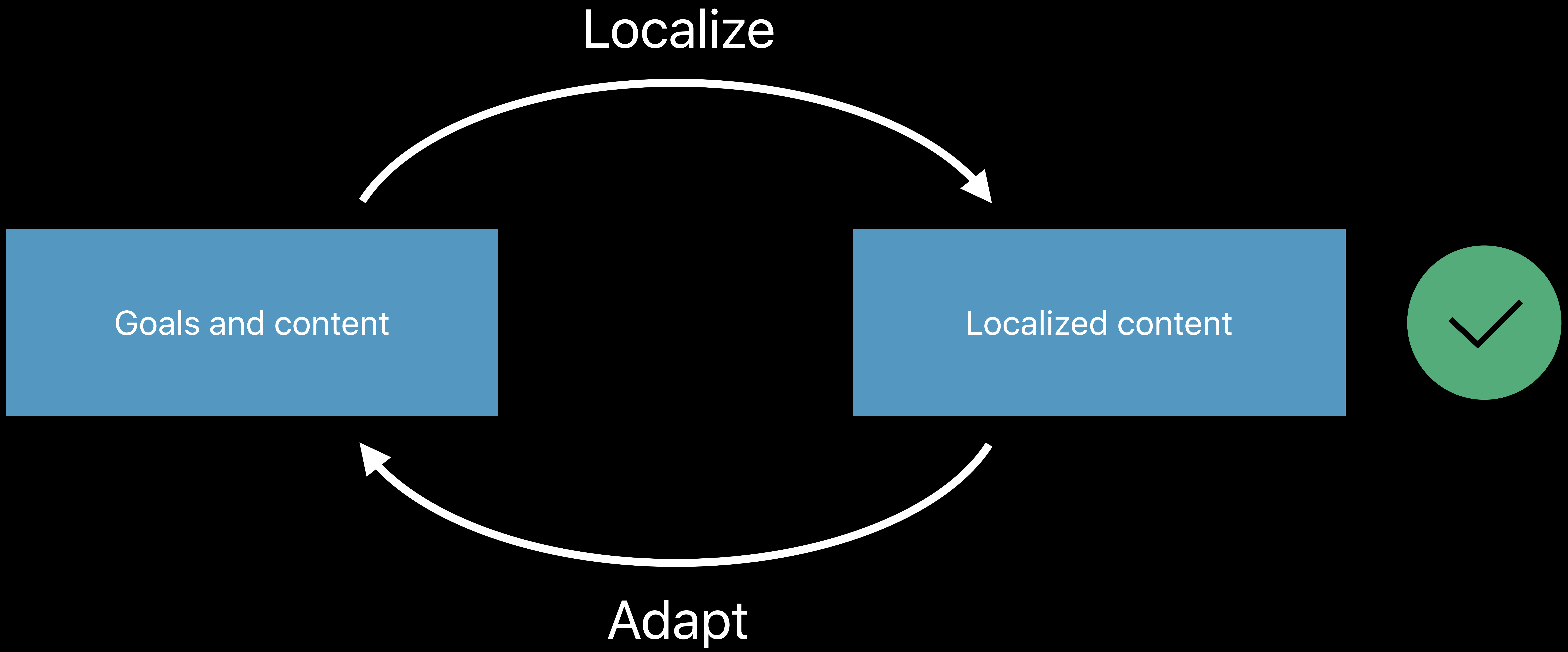
Localize

Goals and content

Localized content

Adapt

Building Localized Content



Localize

Goals and content

Localized content

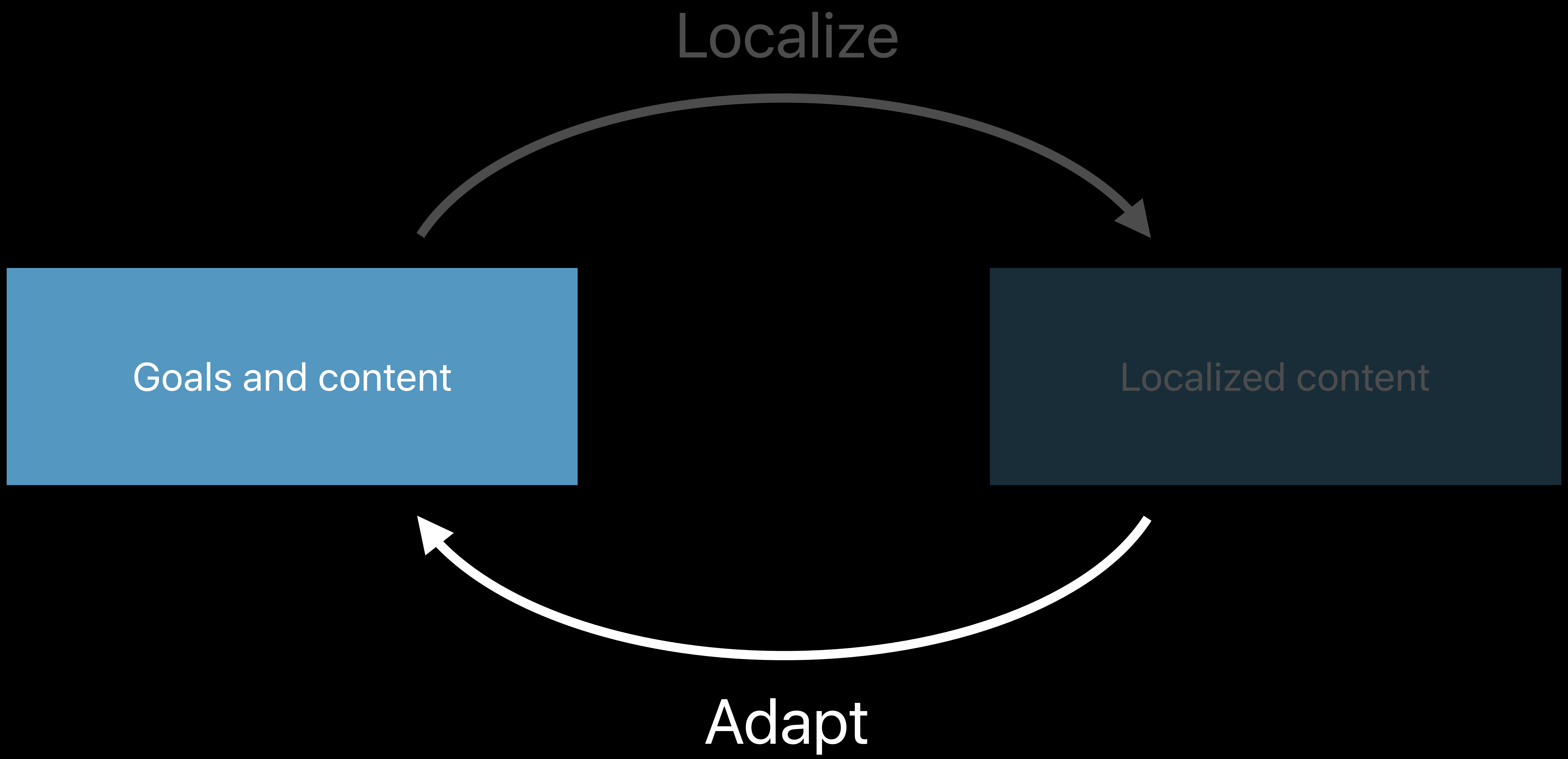
✓

Adapt

Localization Considerations

Aaltan Ahmad, Internationalization Software Engineer

Localization Considerations



Localize

Goals and content

Localized content

Adapt

Content Choices Affect Localizability

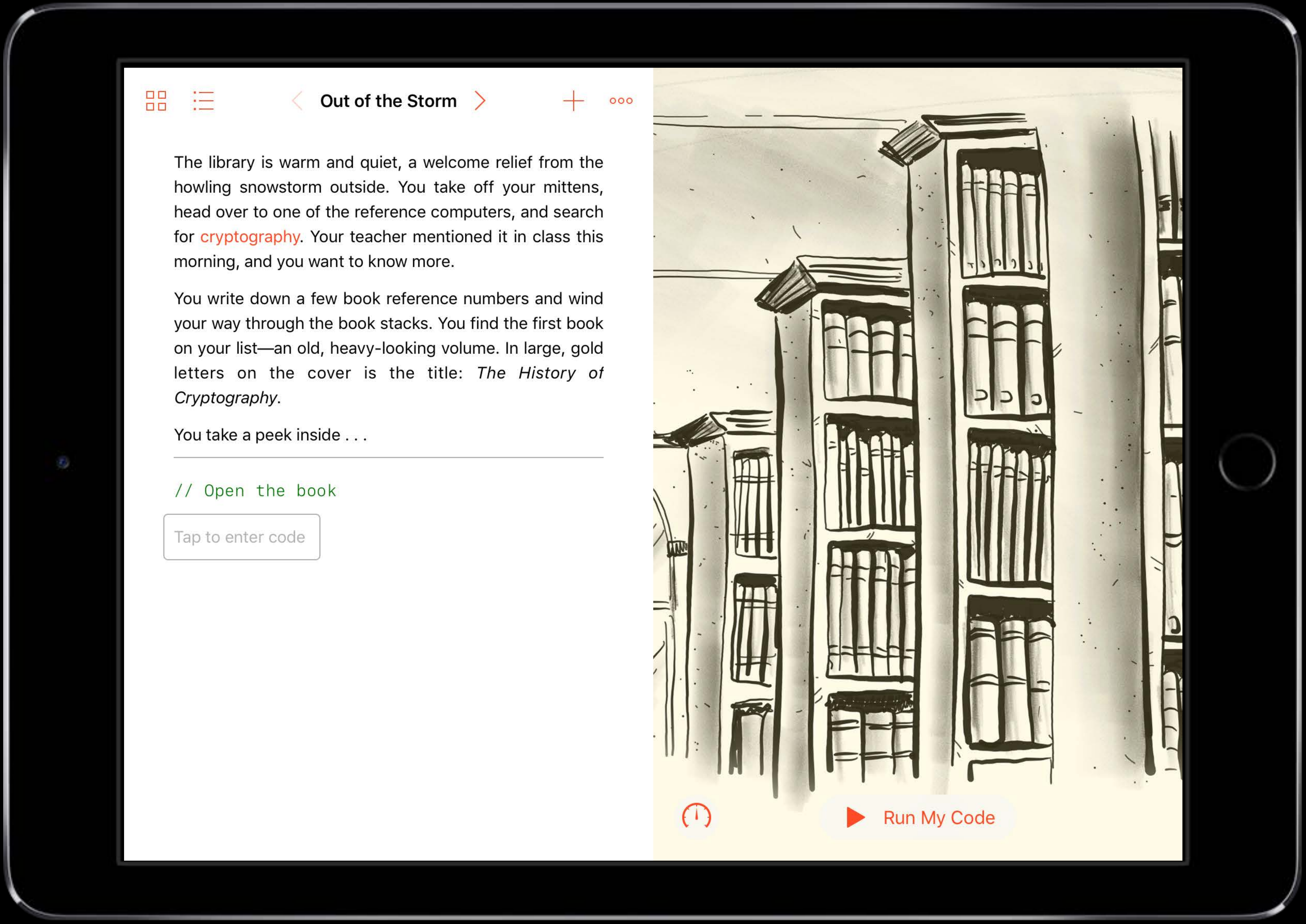
Storyline

Writing style

Language specific concepts

Prose

Storyline



The library is warm and quiet, a welcome relief from the howling snowstorm outside. You take off your mittens, head over to one of the reference computers, and search for cryptography. Your teacher mentioned it in class this morning, and you want to know more.

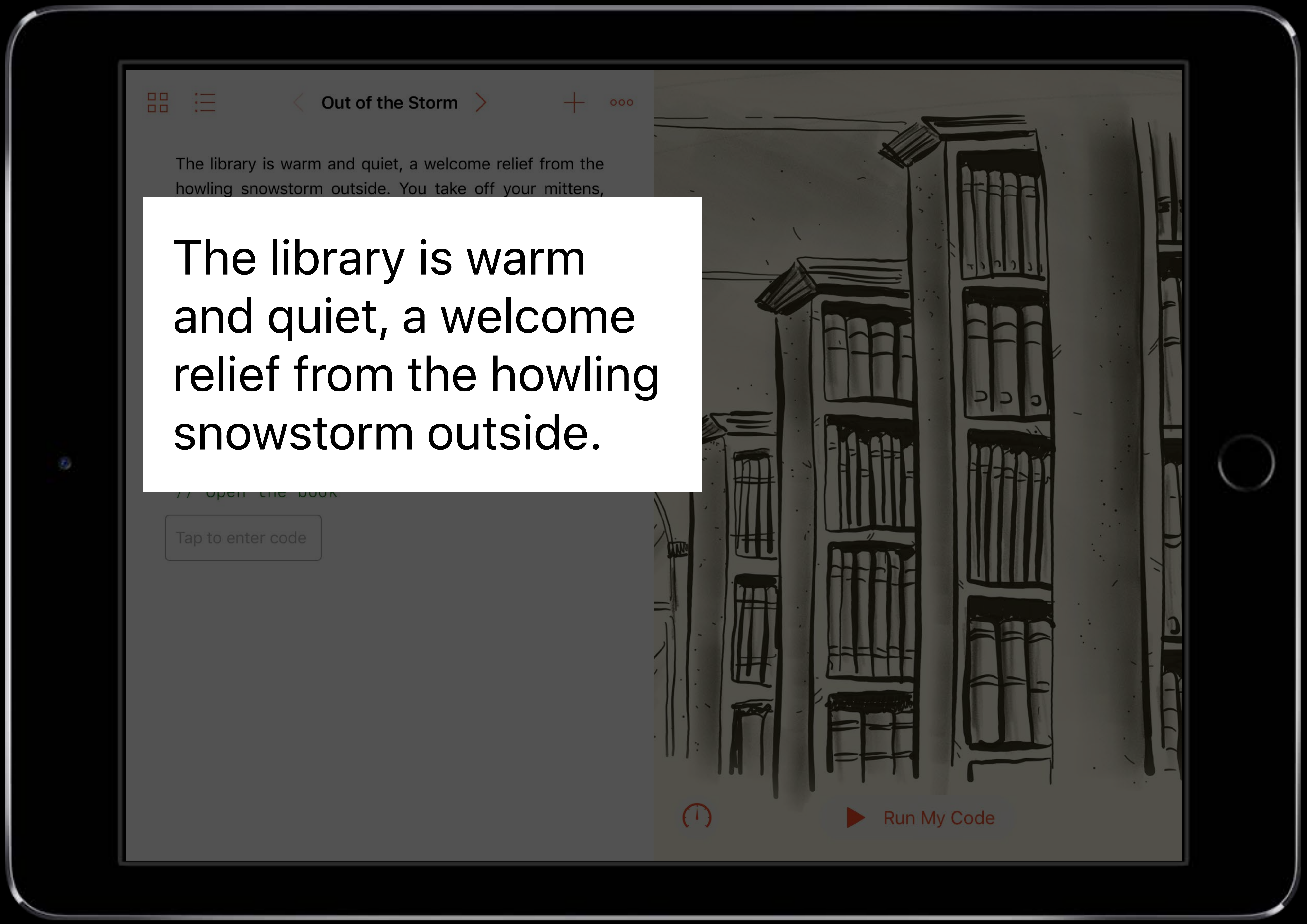
You write down a few book reference numbers and wind your way through the book stacks. You find the first book on your list—an old, heavy-looking volume. In large, gold letters on the cover is the title: *The History of Cryptography*.

You take a peek inside . . .

```
// Open the book
```

Prose

Storyline



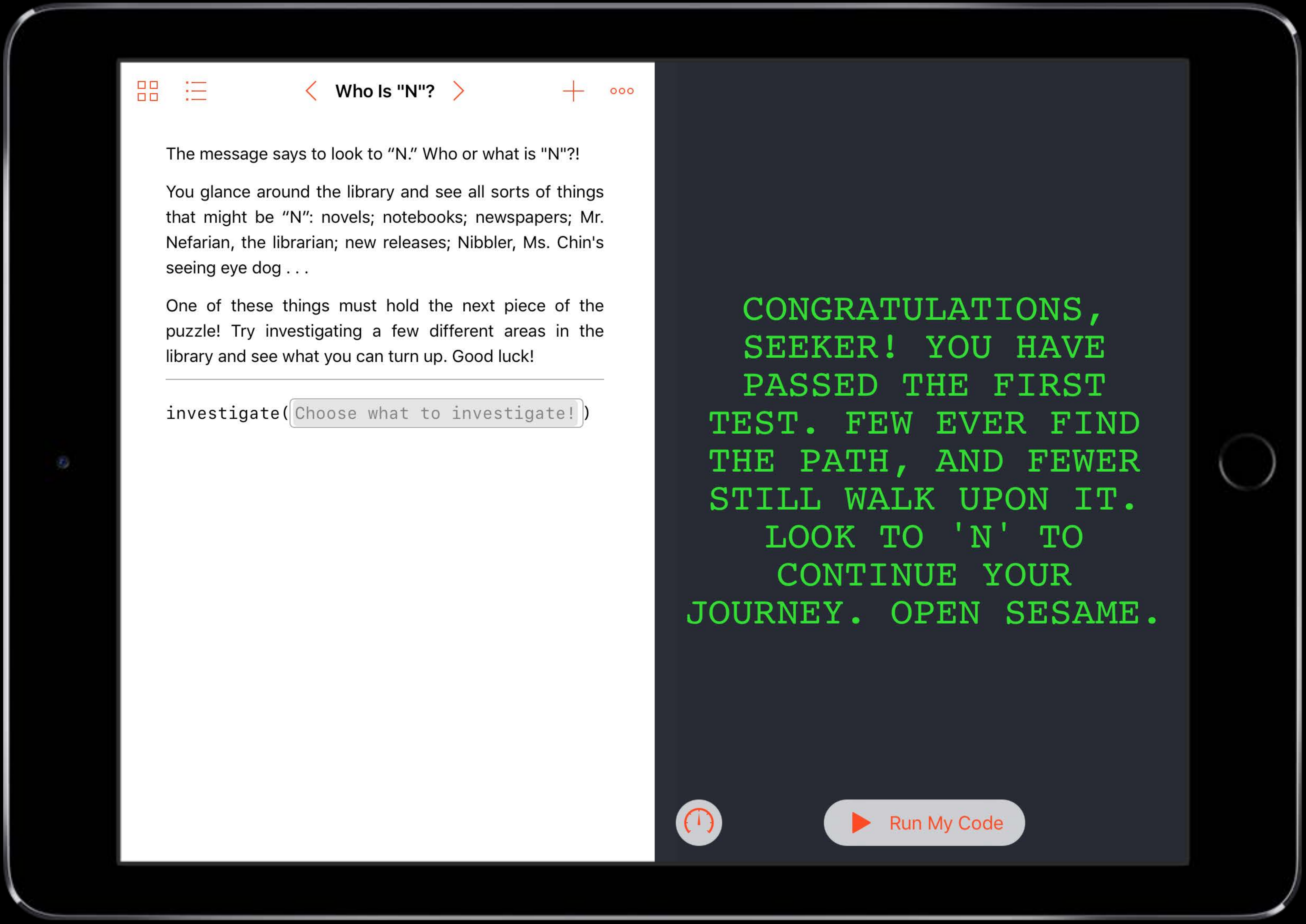
The library is warm and quiet, a welcome relief from the howling snowstorm outside. You take off your mittens,

```
// Open the book
```

The library is warm and quiet, a welcome relief from the howling snowstorm outside.

Prose

Writing style



The message says to look to “N.” Who or what is "N"?!

You glance around the library and see all sorts of things that might be “N”: novels; notebooks; newspapers; Mr. Nefarian, the librarian; new releases; Nibbler, Ms. Chin's seeing eye dog . . .

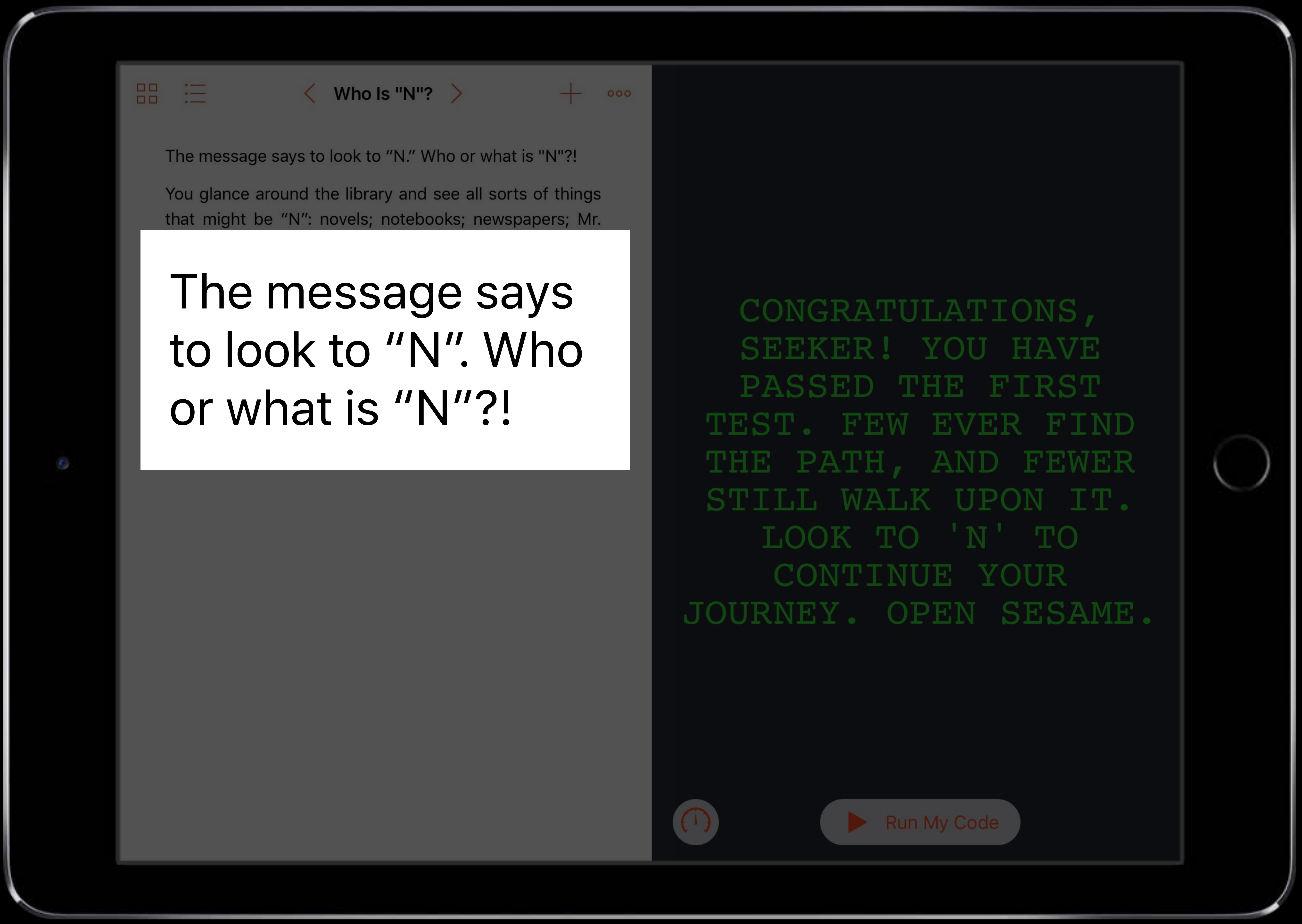
One of these things must hold the next piece of the puzzle! Try investigating a few different areas in the library and see what you can turn up. Good luck!

```
investigate(Choose what to investigate!)
```

```
CONGRATULATIONS,
SEEKER! YOU HAVE
PASSED THE FIRST
TEST. FEW EVER FIND
THE PATH, AND FEWER
STILL WALK UPON IT.
LOOK TO 'N' TO
CONTINUE YOUR
JOURNEY. OPEN SESAME.
```

Prose

Writing style



The message says to look to “N.” Who or what is "N"?!

You glance around the library and see all sorts of things that might be “N”: novels; notebooks; newspapers; Mr.

The message says to look to “N”. Who or what is “N”?!

```
CONGRATULATIONS,
SEEKER! YOU HAVE
PASSED THE FIRST
TEST. FEW EVER FIND
THE PATH, AND FEWER
STILL WALK UPON IT.
LOOK TO 'N' TO
CONTINUE YOUR
JOURNEY. OPEN SESAME.
```

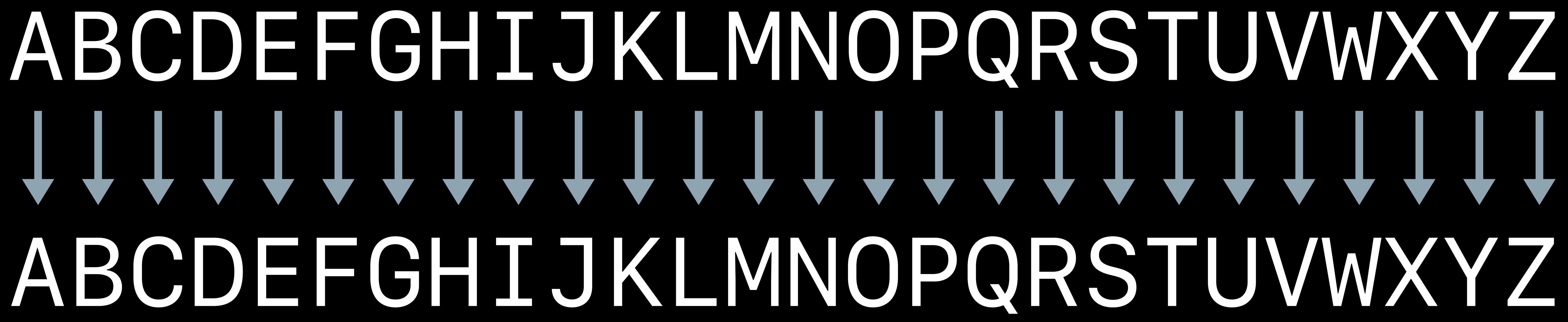
Language Specific Content

Substitution cipher

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Language Specific Content

Substitution cipher

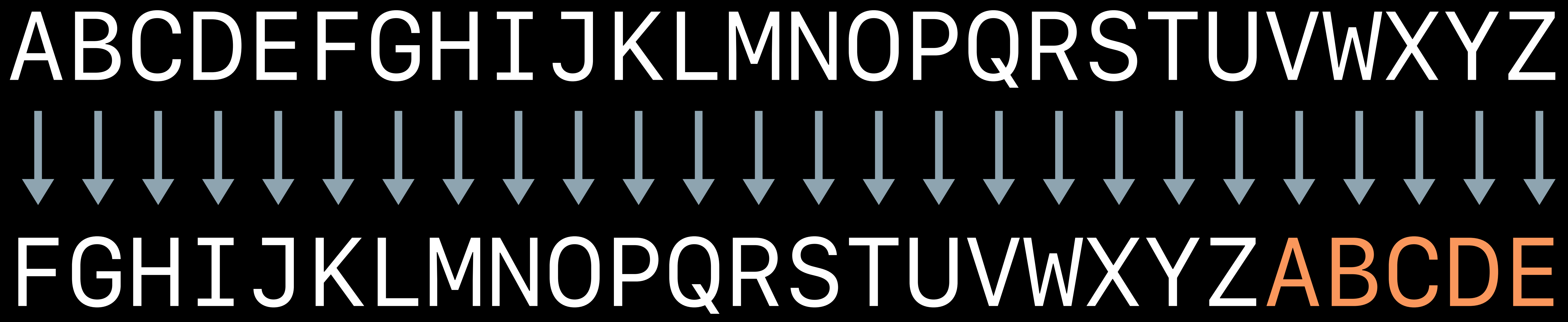


ABCDEFGHIJKLMNOPQRSTUVWXYZ

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Language Specific Content

Substitution cipher

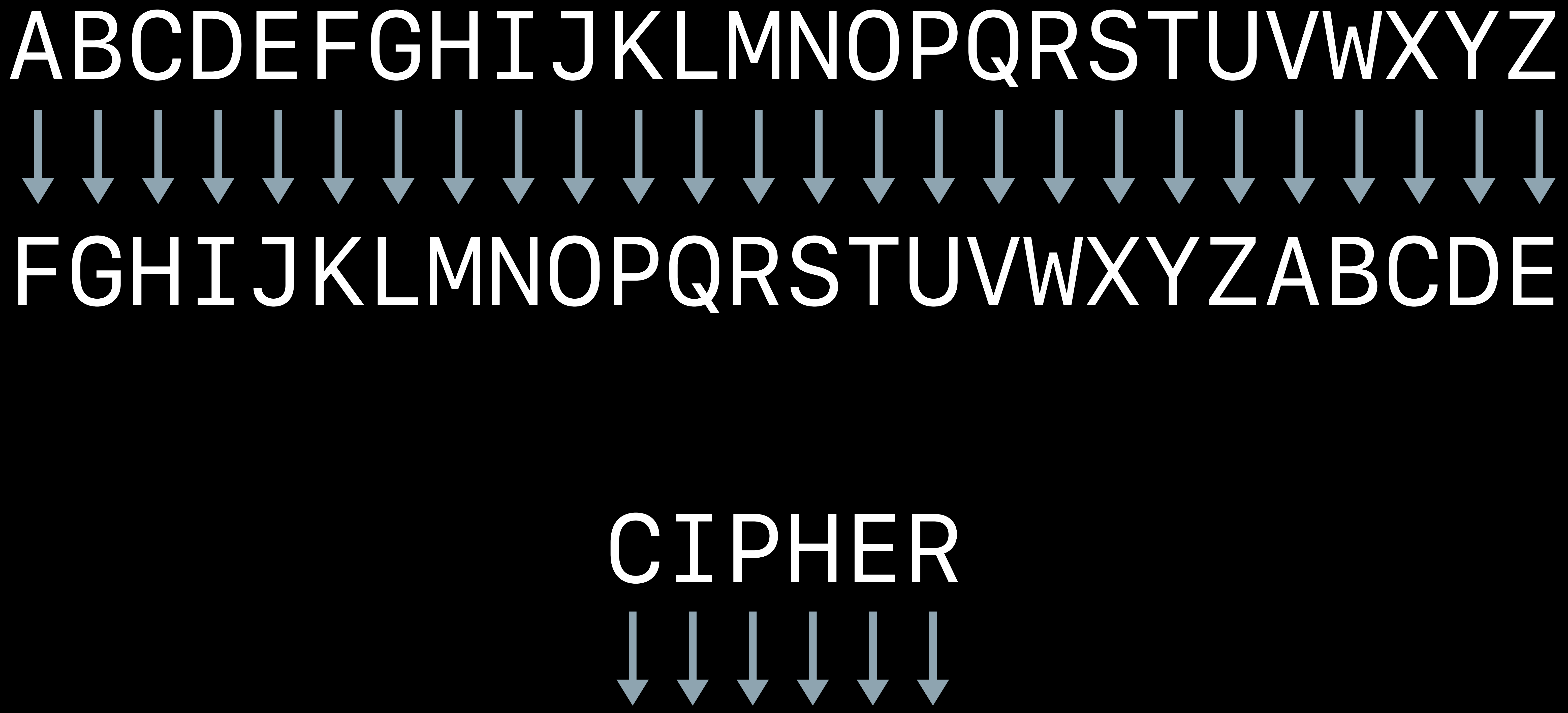


ABCDEFGHIJKLMNOPQRSTUVWXYZ

FGHIJKLMNOPQRSTUVWXYZABCDE

Language Specific Content

Substitution cipher



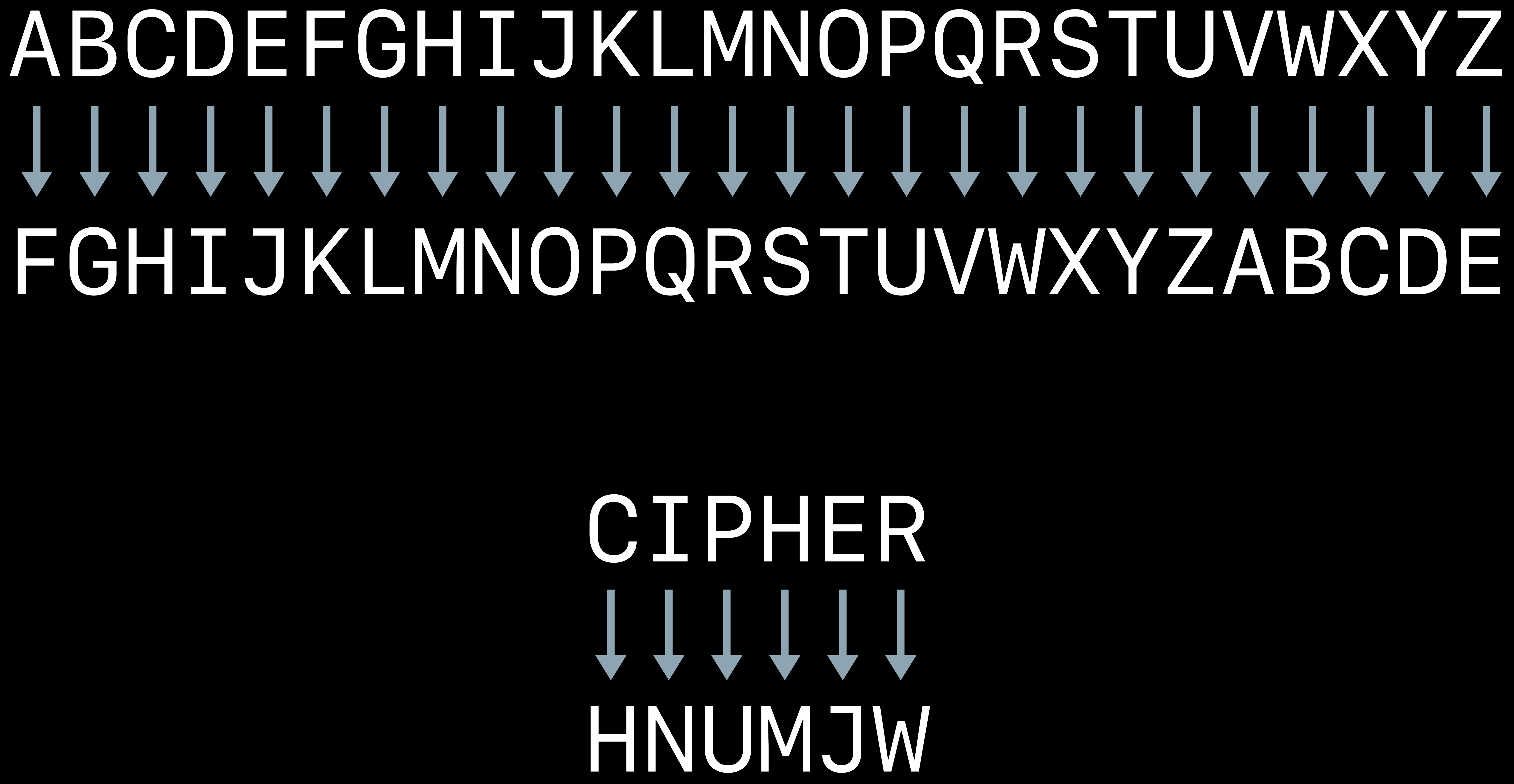
ABCDEFGHIJKLMNOPQRSTUVWXYZ

FGHIJKLMNOPQRSTUVWXYZABCDE

CIPHER

Language Specific Content

Substitution cipher



ABCDEFGHIJKLMNOPQRSTUVWXYZ

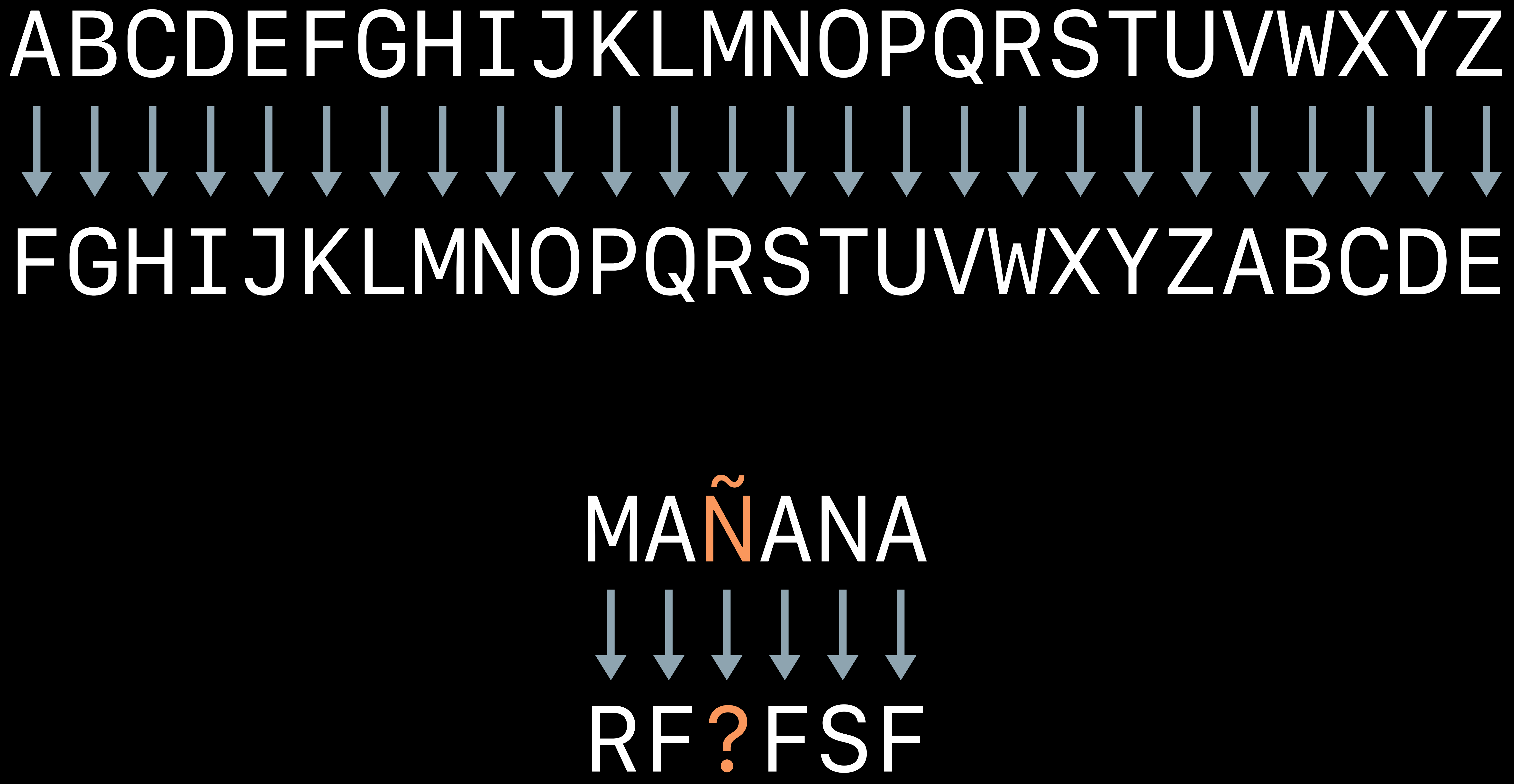
FGHIJKLMNOPQRSTUVWXYZABCDE

CIPHER

HNUMJW

Language Specific Content

Substitution cipher



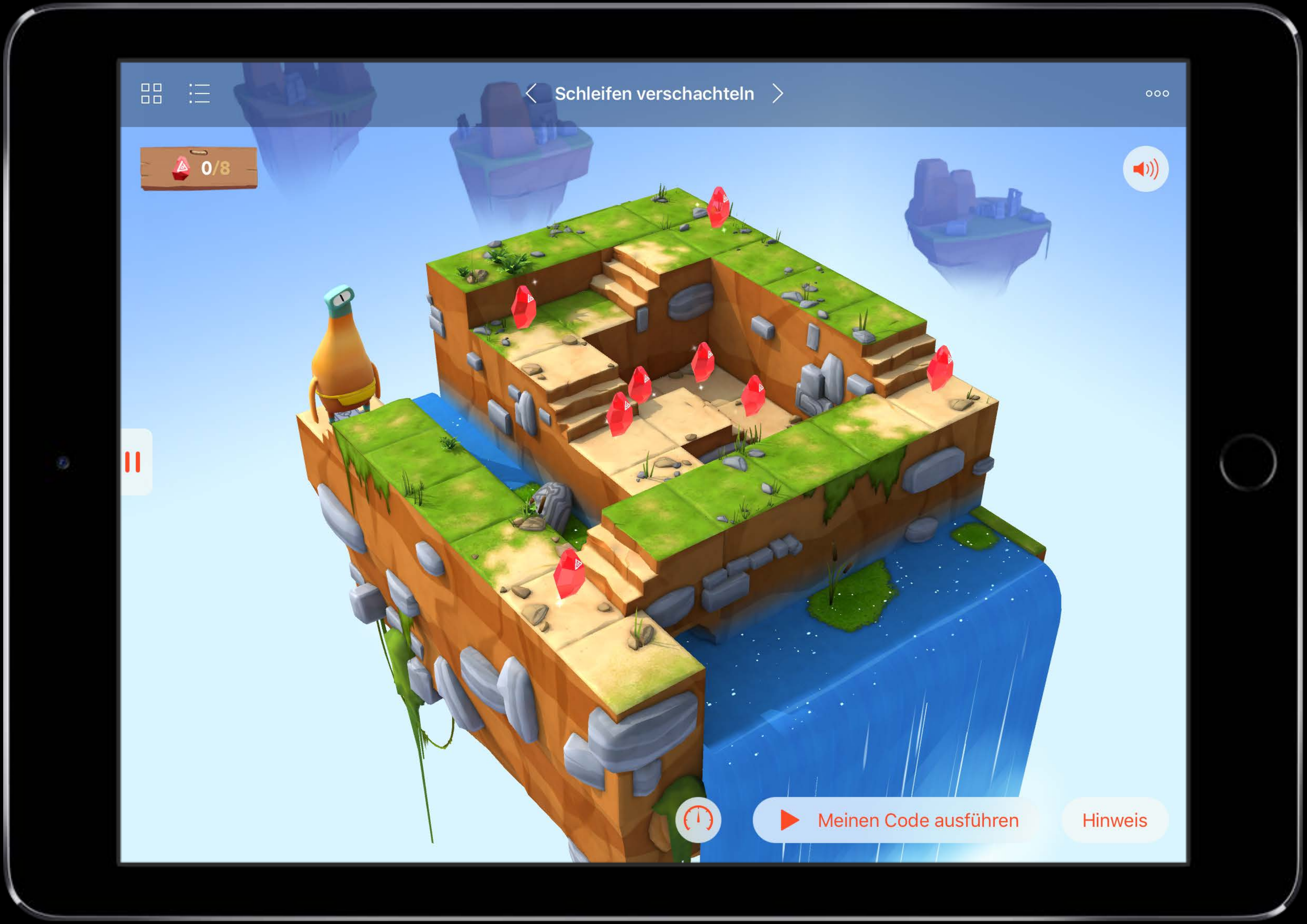
ABCDEFGHIJKLMNOPQRSTUVWXYZ

FGHIJKLMNOPQRSTUVWXYZABCDE

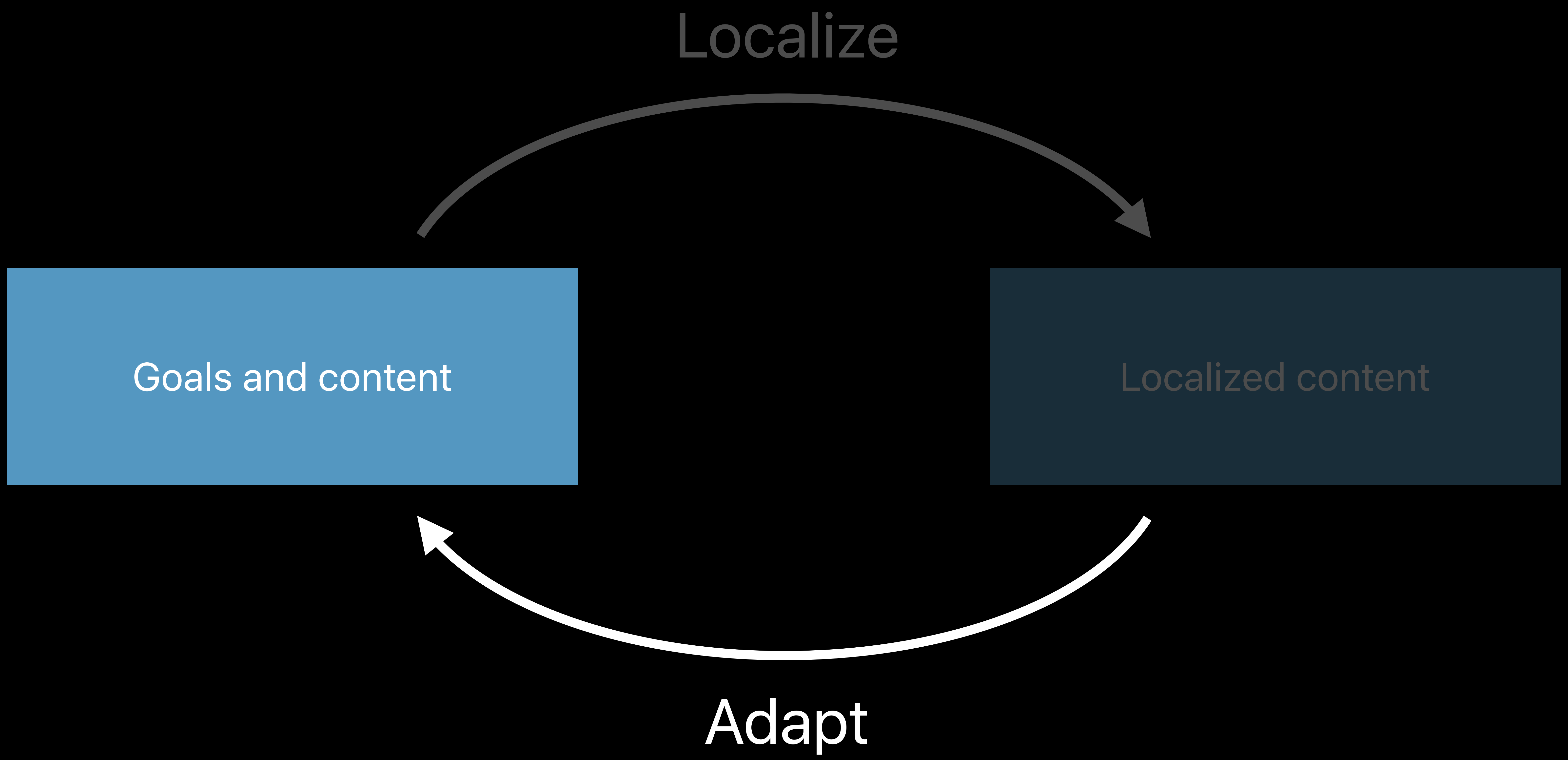
MAÑANA

RF?FSF

Language Agnostic Content



Localization Considerations



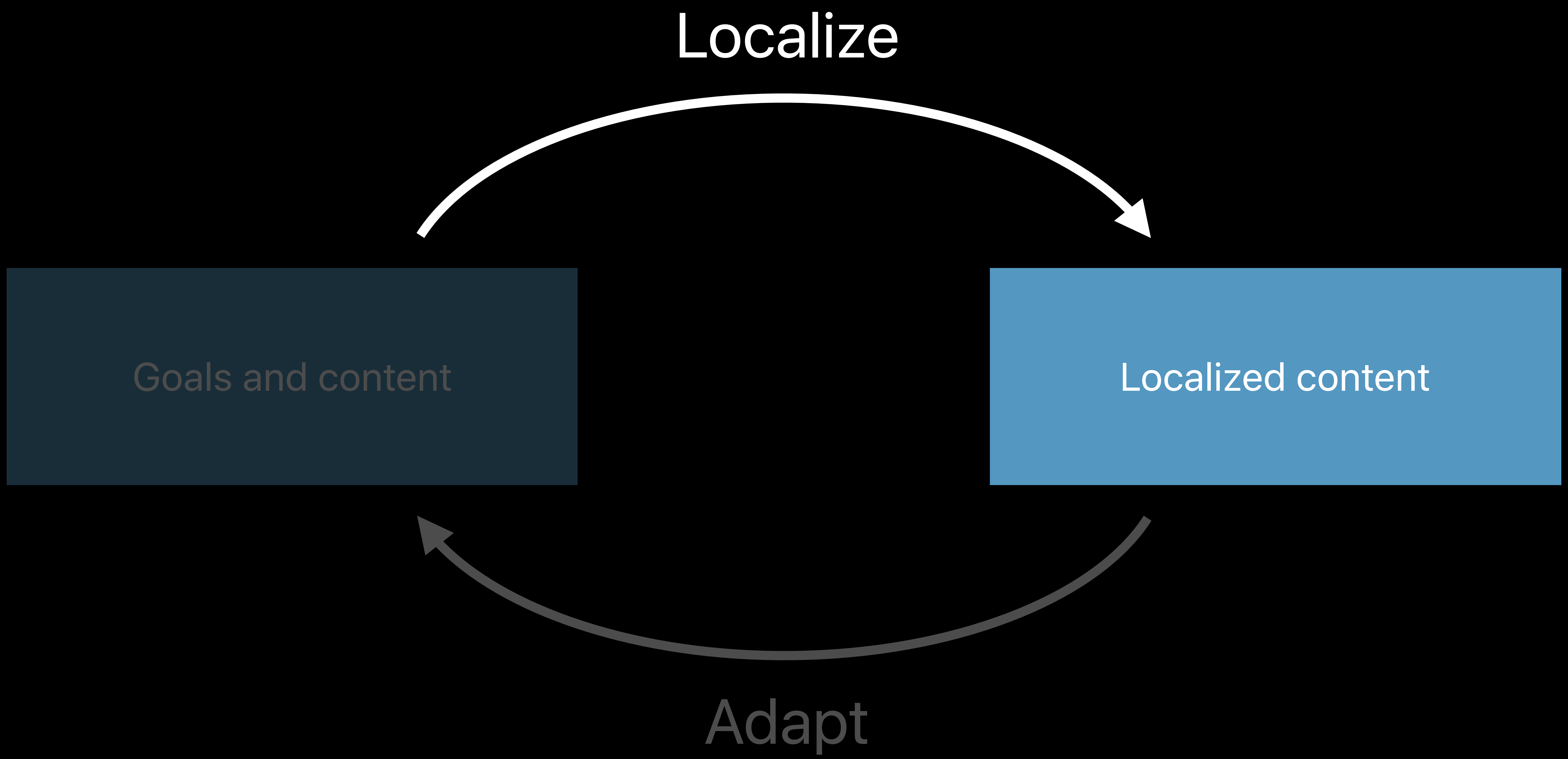
Localize

Goals and content

Localized content

Adapt

Localization Considerations



Localize

Goals and content

Localized content

Adapt

Audience

Formality

Audience

Formality

Language	Informal	Formal
English	you	you

Audience

Formality

Language	Informal	Formal
English	you	you
German	du	Sie

Audience

Formality

Language	Informal	Formal
English	you	you
German	du	Sie
French	tu	vous

Audience

Formality

Language	Informal	Formal
English	you	you
German	du	Sie
French	tu	vous

Audience

Formality

Audience

Formality

	Formality	Example
iOS	Formal	Connectez-vous avec votre identifiant
Swift Playgrounds	Informal	Tu devras écrire des commandes

Audience

Age

Audience

Age

kan ji

漢字

Audience

Age

English	Standard Japanese	Child-friendly Japanese
code area	コード領域 (code sphere/territory)	コードを書く場所 (place to write code)

Technical Terms

Technical Terms

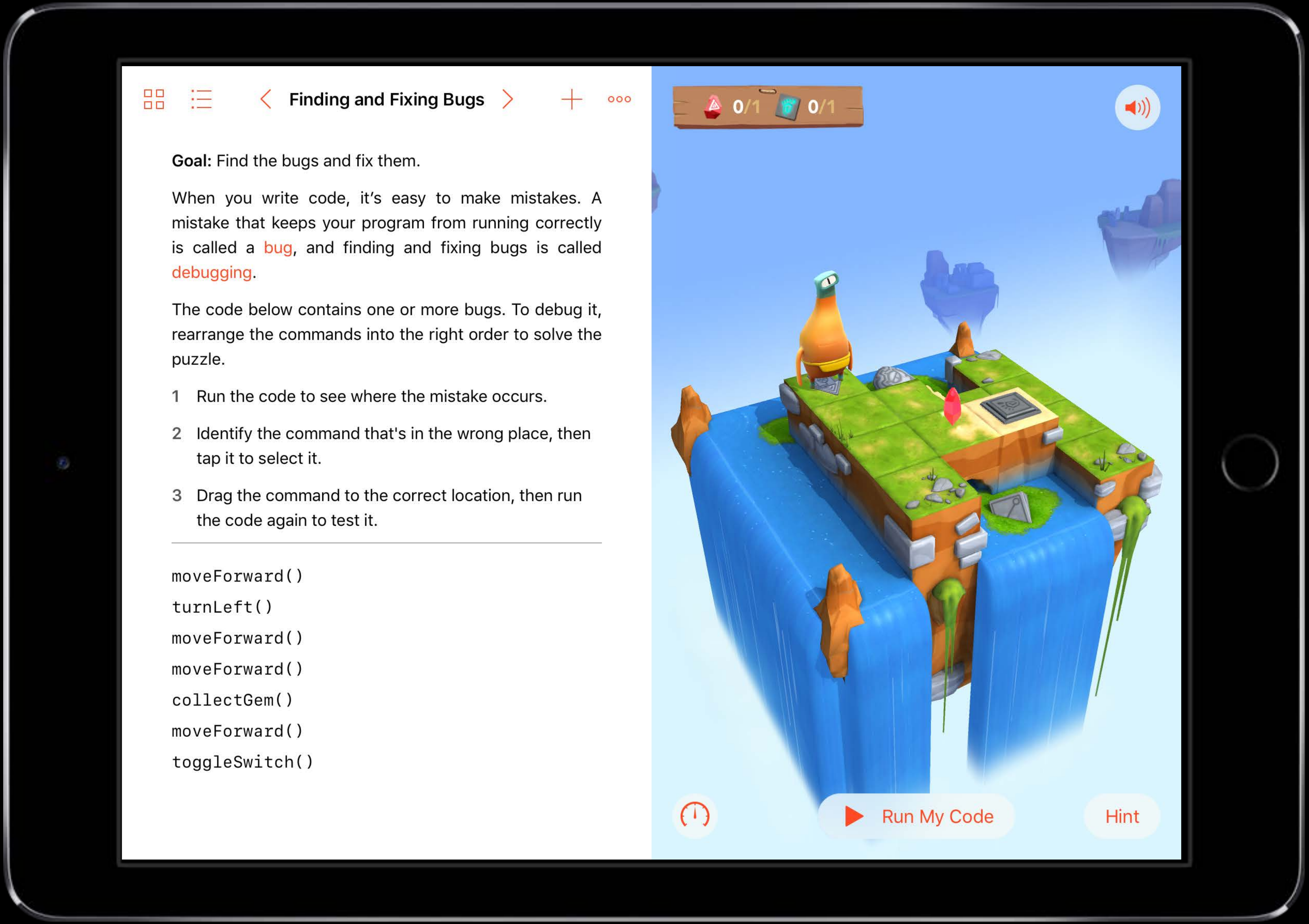
Computer Science terms

Mathematical terms

Swift keywords, enums, and functions

Technical Terms

Introducing “bug”



Goal: Find the bugs and fix them.

When you write code, it’s easy to make mistakes. A mistake that keeps your program from running correctly is called a bug, and finding and fixing bugs is called debugging.

The code below contains one or more bugs. To debug it, rearrange the commands into the right order to solve the puzzle.

1 Run the code to see where the mistake occurs.

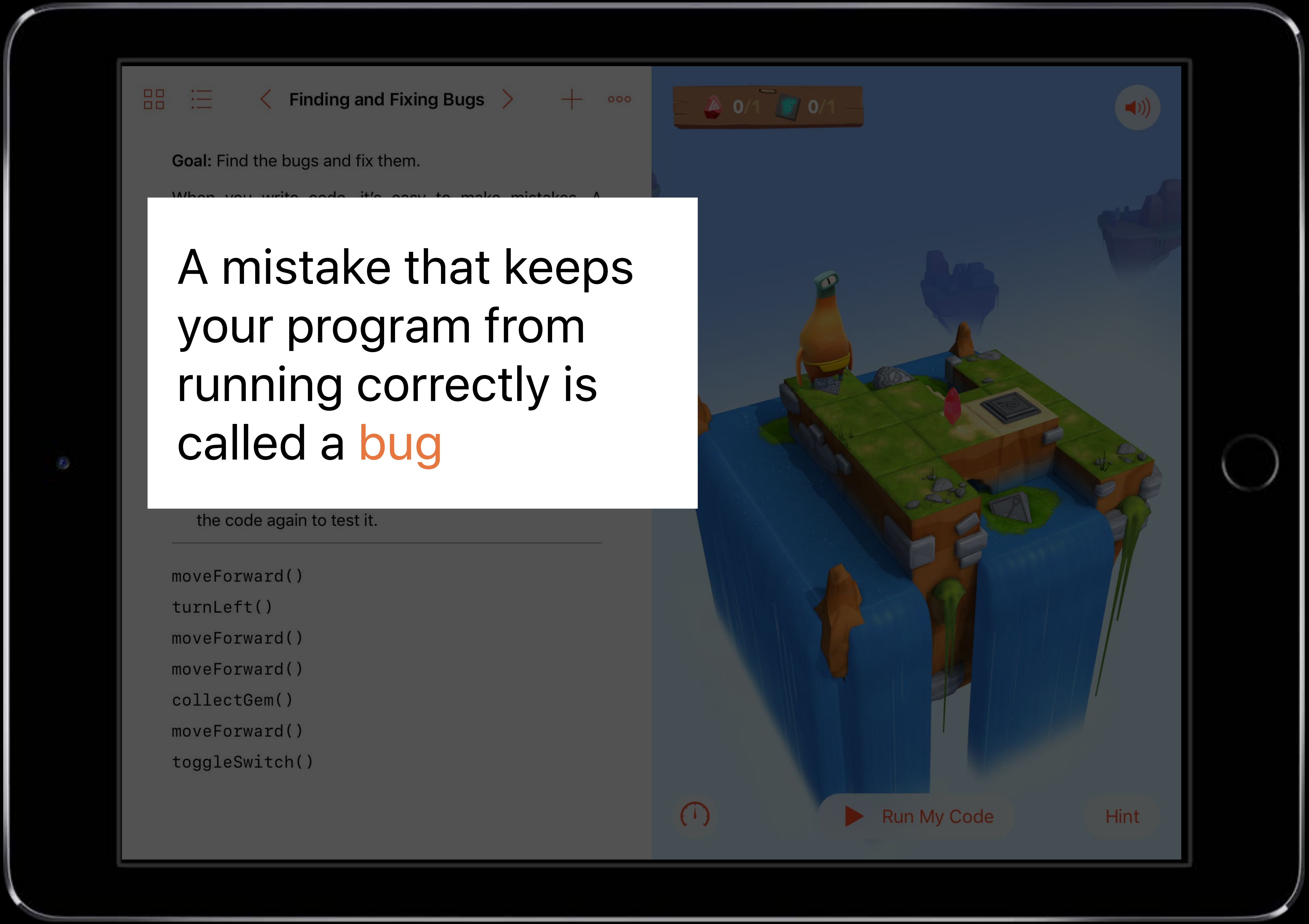
2 Identify the command that's in the wrong place, then tap it to select it.

3 Drag the command to the correct location, then run the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

Technical Terms

Introducing "bug"



A mistake that keeps your program from running correctly is called a bug

Goal: Find the bugs and fix them.

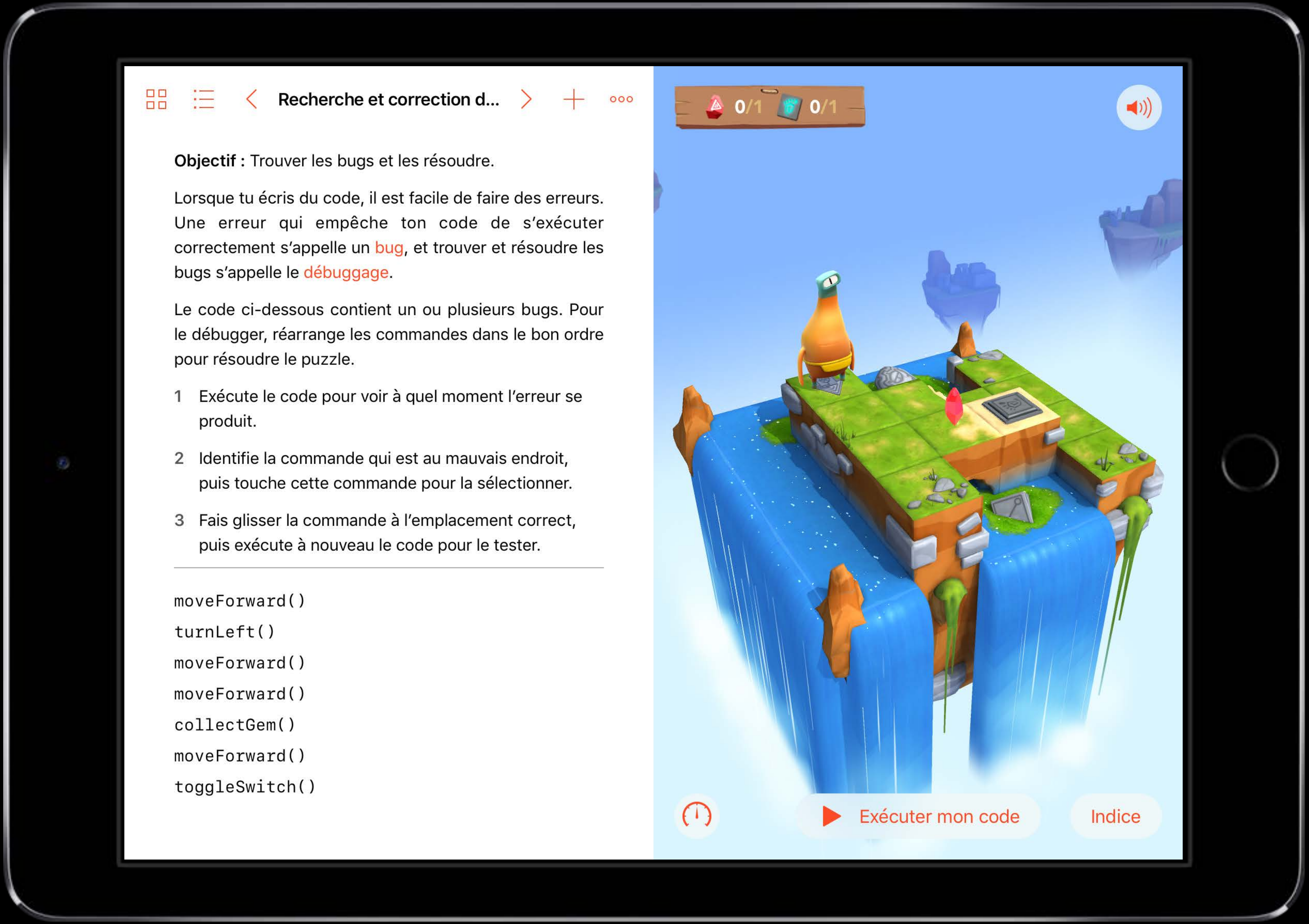
When you write code, it's easy to make mistakes. A

the code again to test it.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

Technical Terms

Introducing "bug"



Objectif : Trouver les bugs et les résoudre.

Lorsque tu écris du code, il est facile de faire des erreurs. Une erreur qui empêche ton code de s'exécuter correctement s'appelle un bug, et trouver et résoudre les bugs s'appelle le débuggage.

Le code ci-dessous contient un ou plusieurs bugs. Pour le débugger, réarrange les commandes dans le bon ordre pour résoudre le puzzle.

1 Exécute le code pour voir à quel moment l'erreur se produit.

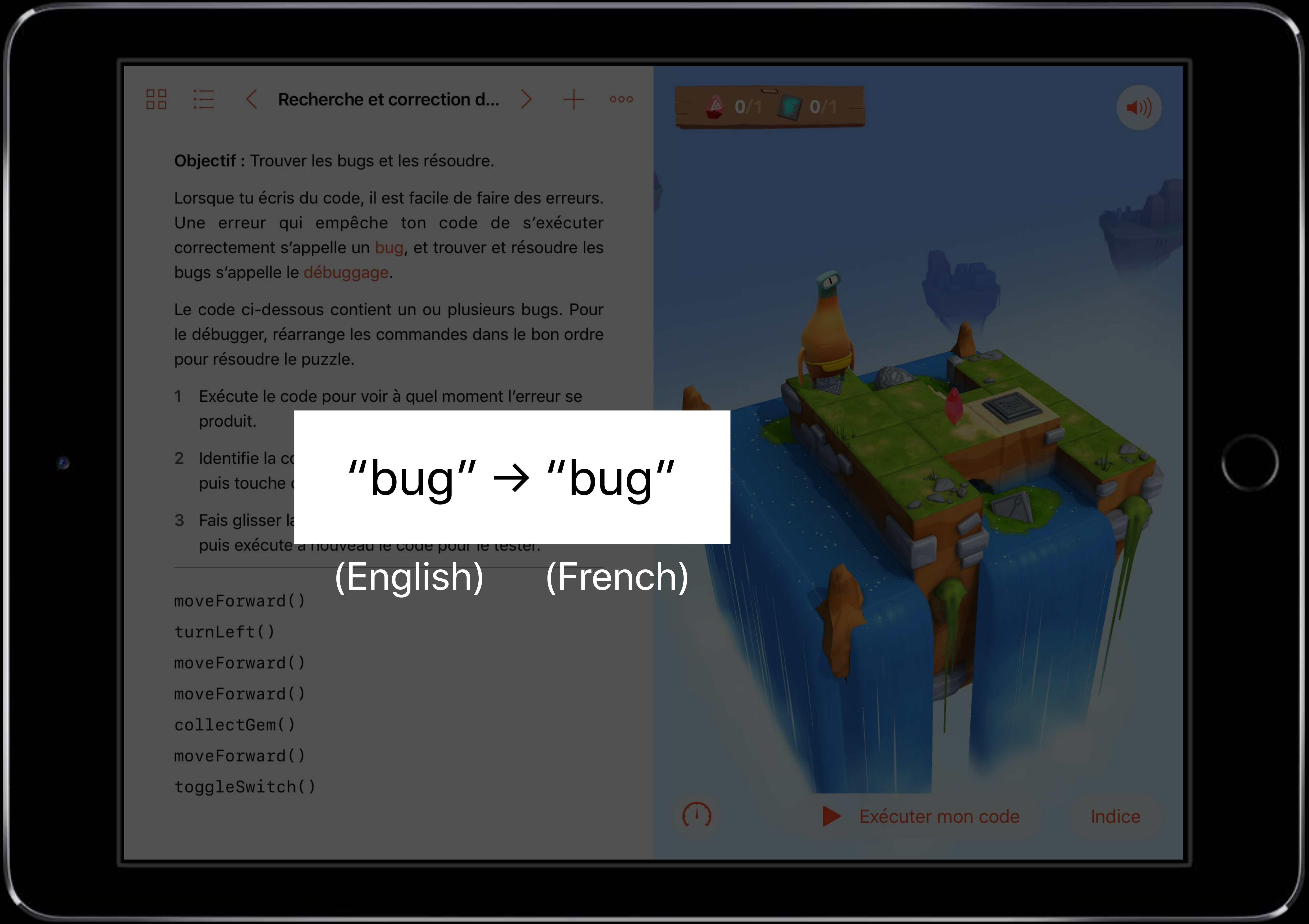
2 Identifie la commande qui est au mauvais endroit, puis touche cette commande pour la sélectionner.

3 Fais glisser la commande à l'emplacement correct, puis exécute à nouveau le code pour le tester.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

Technical Terms

Introducing “bug”



Recherche et correction d...

Objectif : Trouver les bugs et les résoudre.

Lorsque tu écris du code, il est facile de faire des erreurs. Une erreur qui empêche ton code de s’exécuter correctement s’appelle un bug, et trouver et résoudre les bugs s’appelle le débuggage.

Le code ci-dessous contient un ou plusieurs bugs. Pour le débugger, réarrange les commandes dans le bon ordre pour résoudre le puzzle.

1 Exécute le code pour voir à quel moment l’erreur se produit.

2 Identifie la c
puis touche

3 Fais glisser l
puis exécute à nouveau le code pour le tester.

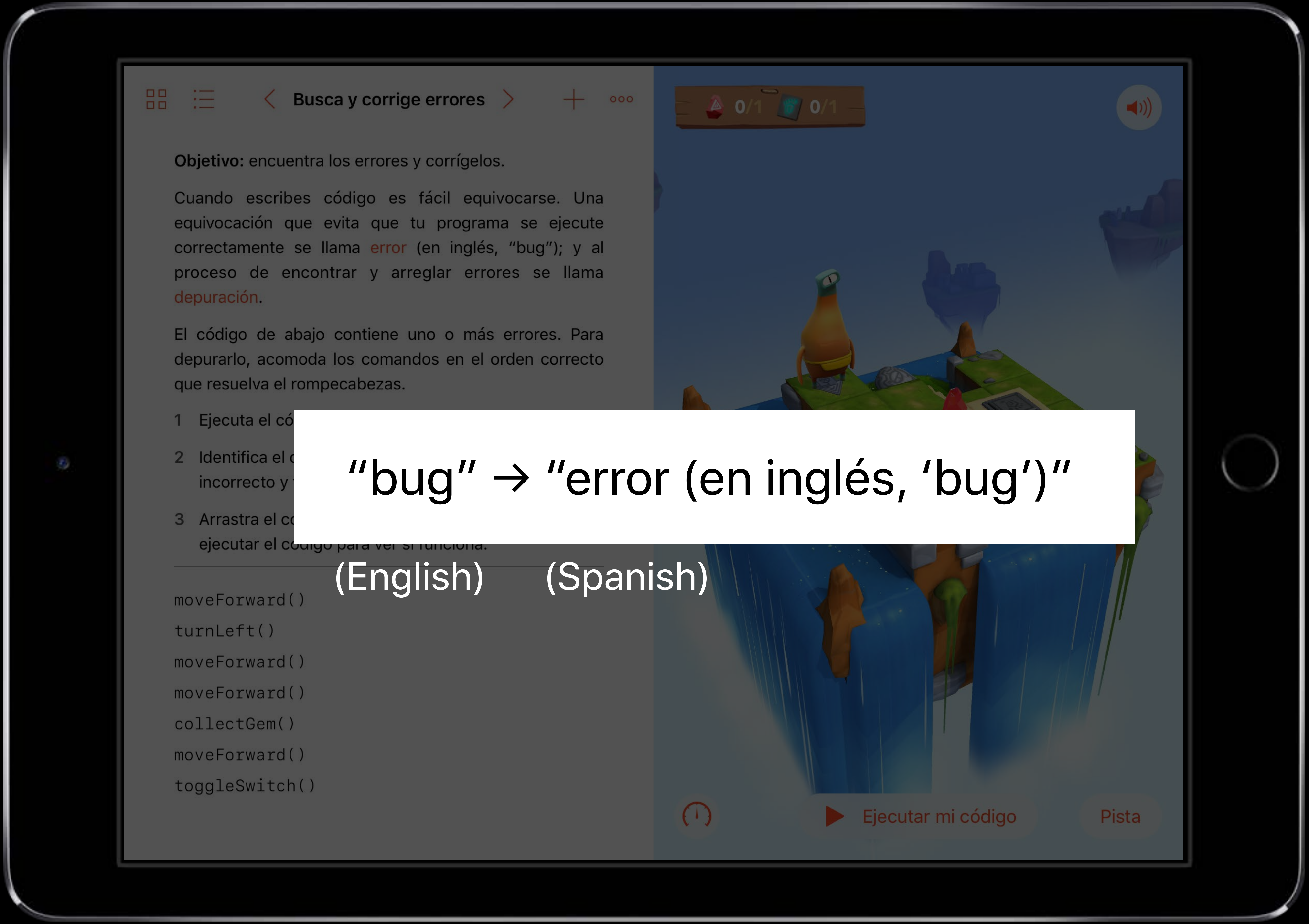
```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

“bug” → “bug”

(English) (French)

Technical Terms

Introducing “bug”



Objetivo: encuentra los errores y corrígelos.

Cuando escribes código es fácil equivocarse. Una equivocación que evita que tu programa se ejecute correctamente se llama error (en inglés, “bug”); y al proceso de encontrar y arreglar errores se llama depuración.

El código de abajo contiene uno o más errores. Para depurarlo, acomoda los comandos en el orden correcto que resuelva el rompecabezas.

1 Ejecuta el có

2 Identifica el c
incorrecto y

3 Arrastra el c
ejecutar el código para ver si funciona.

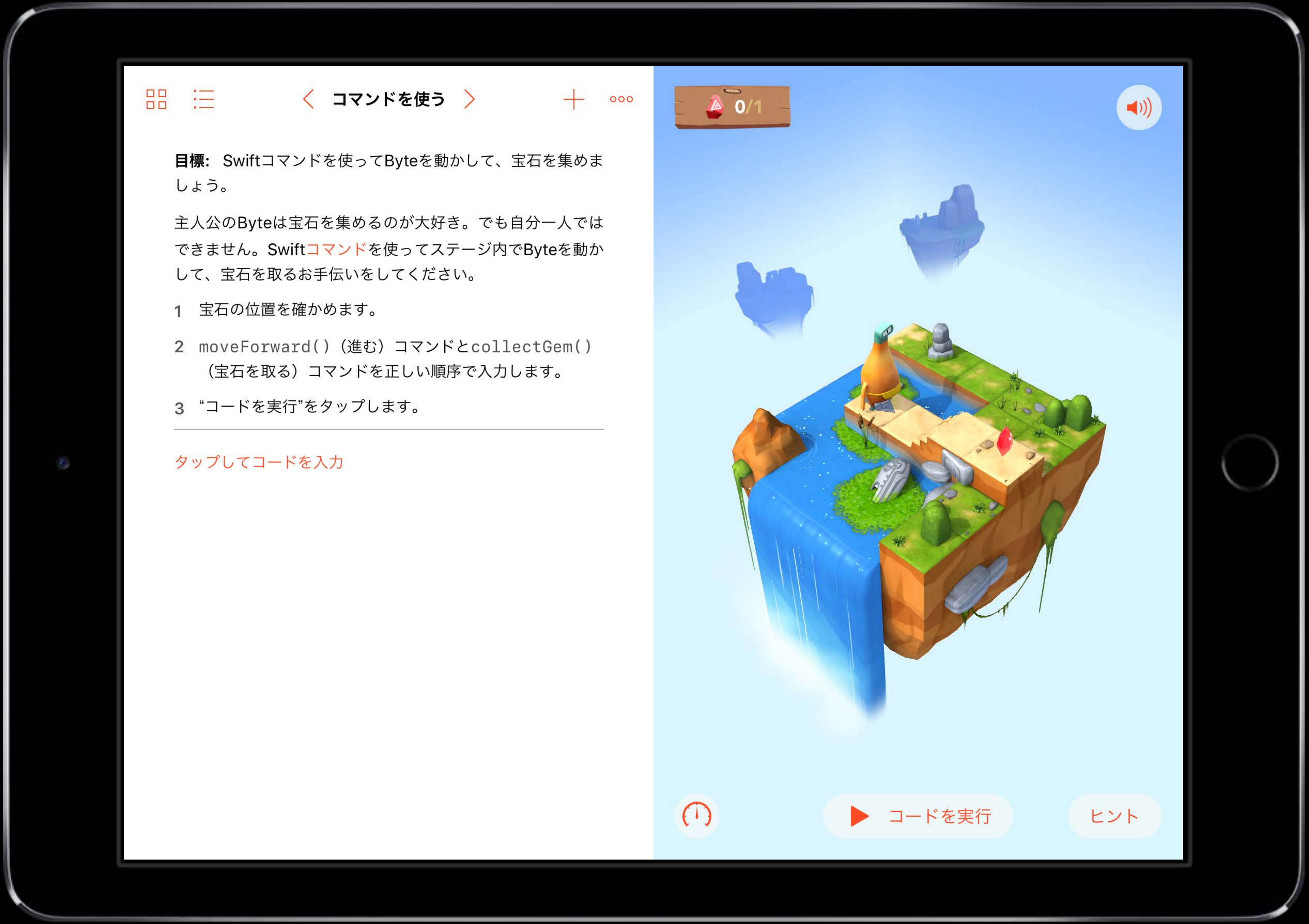
```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

“bug” → “error (en inglés, ‘bug’)”

(English) (Spanish)

Technical Terms

Encountering English



コマンドを使う

目標: SwiftコマンドをBytesを動かして、宝石を集めましょう。

主人公のByteは宝石を集めるのが大好き。でも自分一人ではできません。Swiftコマンドを使ってステージ内でByteを動かして、宝石を取るお手伝いをしてください。

1 宝石の位置を確かめます。

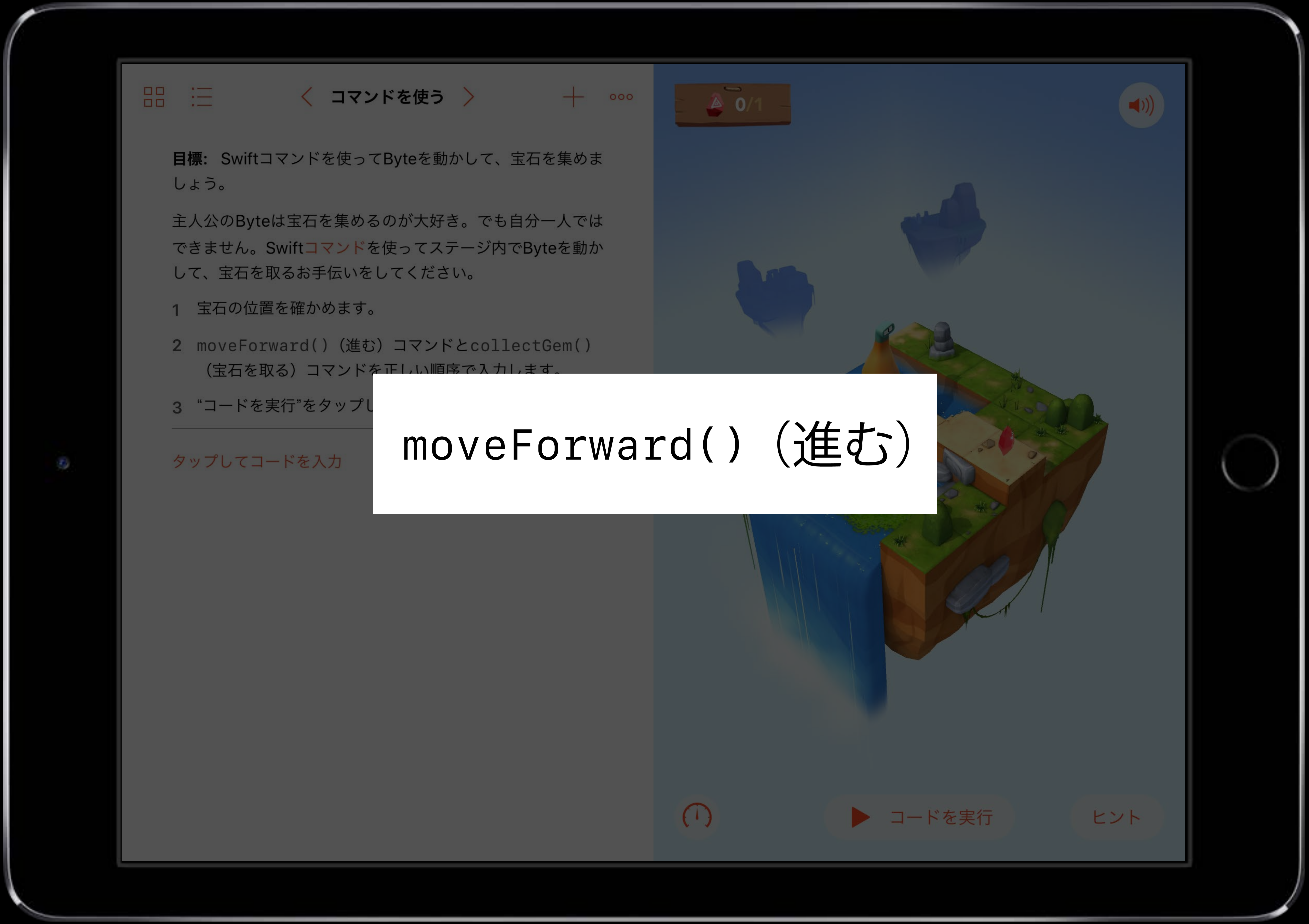
2 moveForward()（進む）コマンドとcollectGem()（宝石を取る）コマンドを正しい順序で入力します。

3 “コードを実行”をタップします。

タップしてコードを入力

Technical Terms

Encountering English



目標: Swiftコマンドを使ってByteを動かして、宝石を集めましょう。

主人公のByteは宝石を集めるのが大好き。でも自分一人ではできません。Swiftコマンドを使ってステージ内でByteを動かして、宝石を取るお手伝いをしてください。

1 宝石の位置を確かめます。

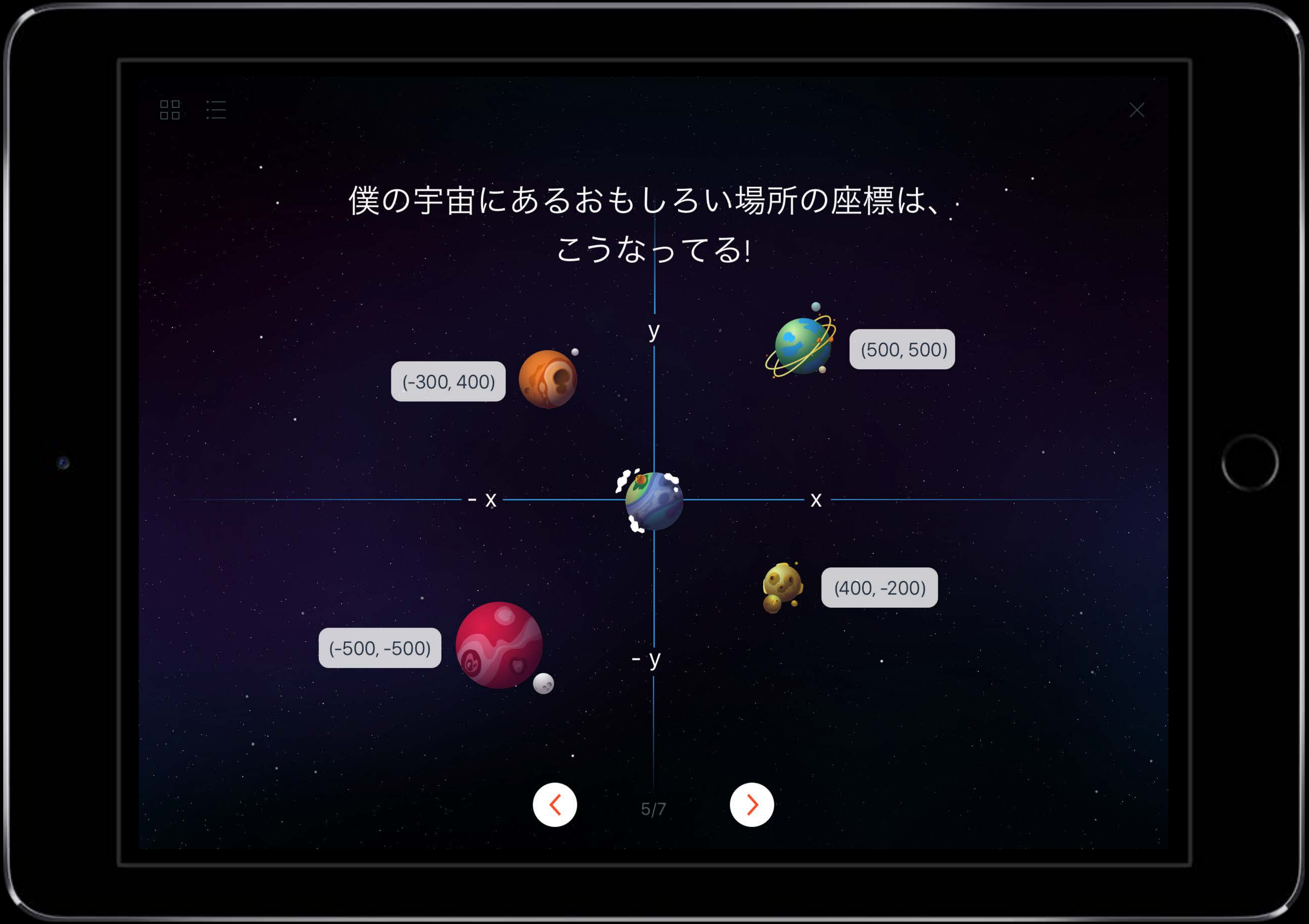
2 moveForward()（進む）コマンドとcollectGem()（宝石を取る）コマンドを正しい順序で入力します。

3 “コードを実行”をタップし

moveForward()（進む）

Technical Terms

Editorial freedom



僕の宇宙にあるおもしろい場所の座標は、
こうなってる!

Technical Terms

Editorial freedom

English	Japanese
four quarters, or *quadrants*	空間（象限…） (area) (quadrant)

Technical Terms

Technical Terms

Unfamiliar terms can be confusing

Technical Terms

Unfamiliar terms can be confusing

English comfort varies

Technical Terms

Unfamiliar terms can be confusing

English comfort varies

Consistency matters

Things to Remember

Things to Remember

Content goals and audience

Things to Remember

Content goals and audience

Linguistic variations

Things to Remember

Content goals and audience

Linguistic variations

Style guide

Things to Remember

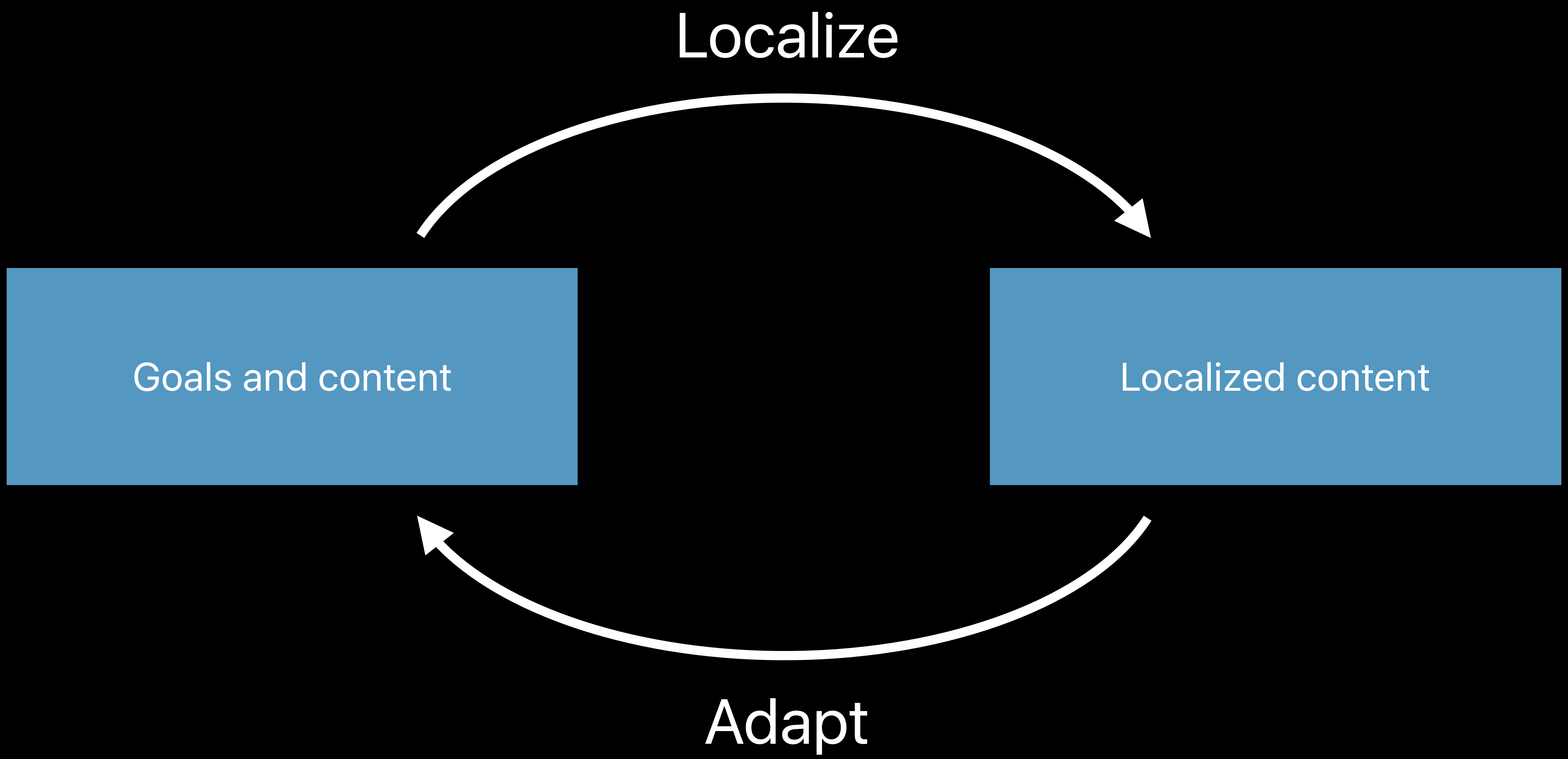
Content goals and audience

Linguistic variations

Style guide

User testing

Things to Remember



Localize

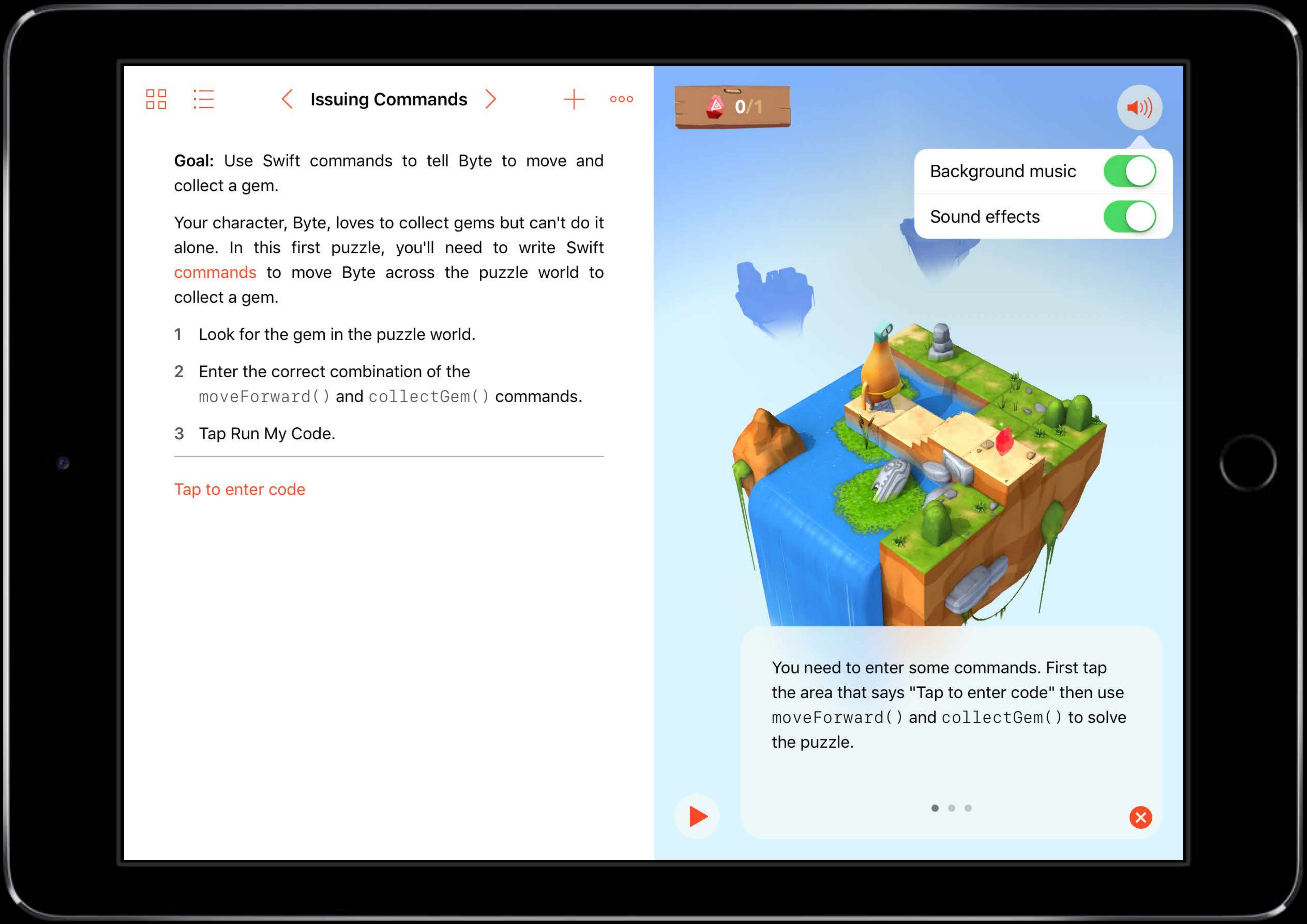
Goals and content

Localized content

Adapt

Structure of a Localized Playground Book

Alex Kuhn, Playgrounds Content Developer



Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

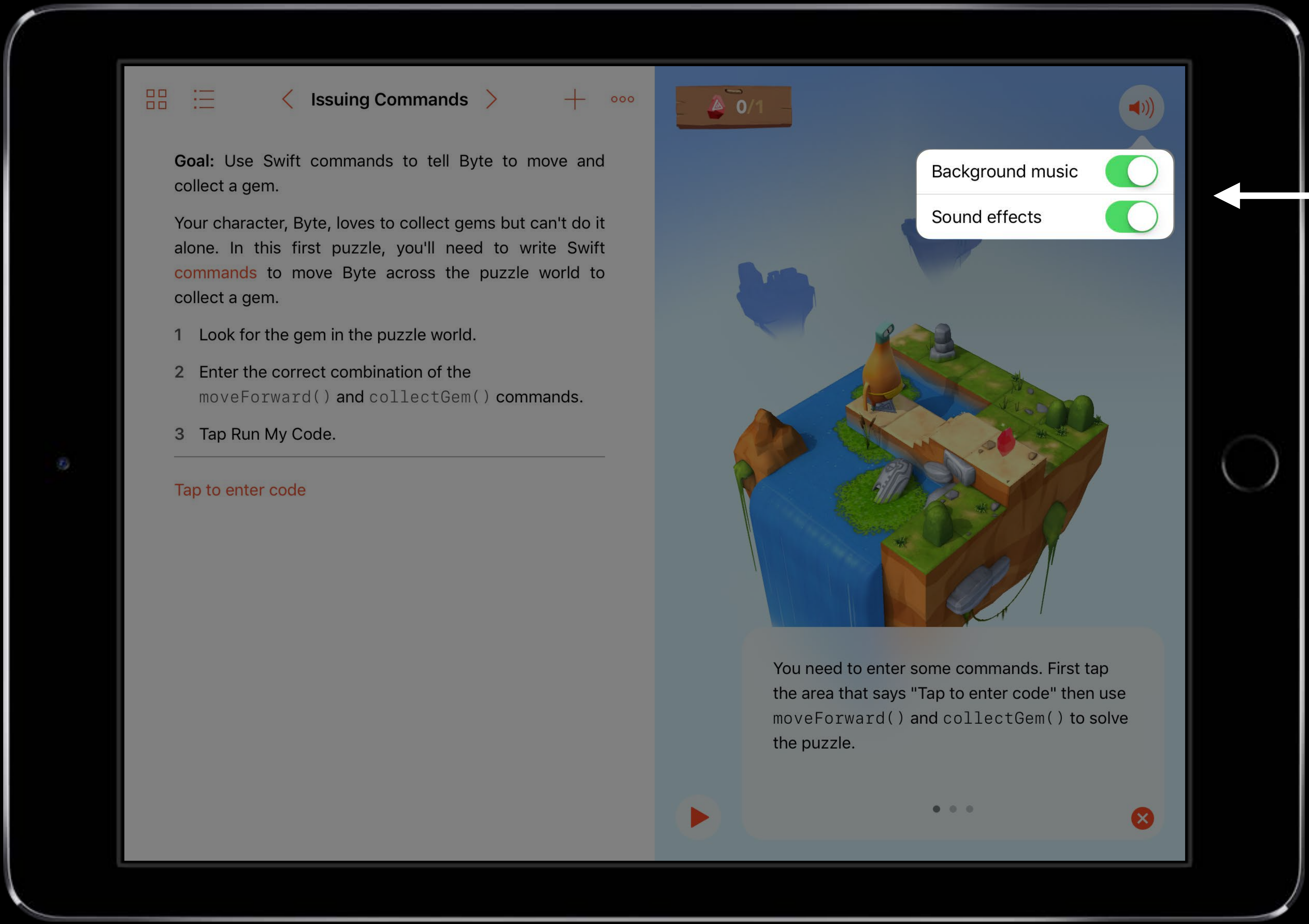
3 Tap Run My Code.

Tap to enter code

Background music ●

Sound effects ●

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.



Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

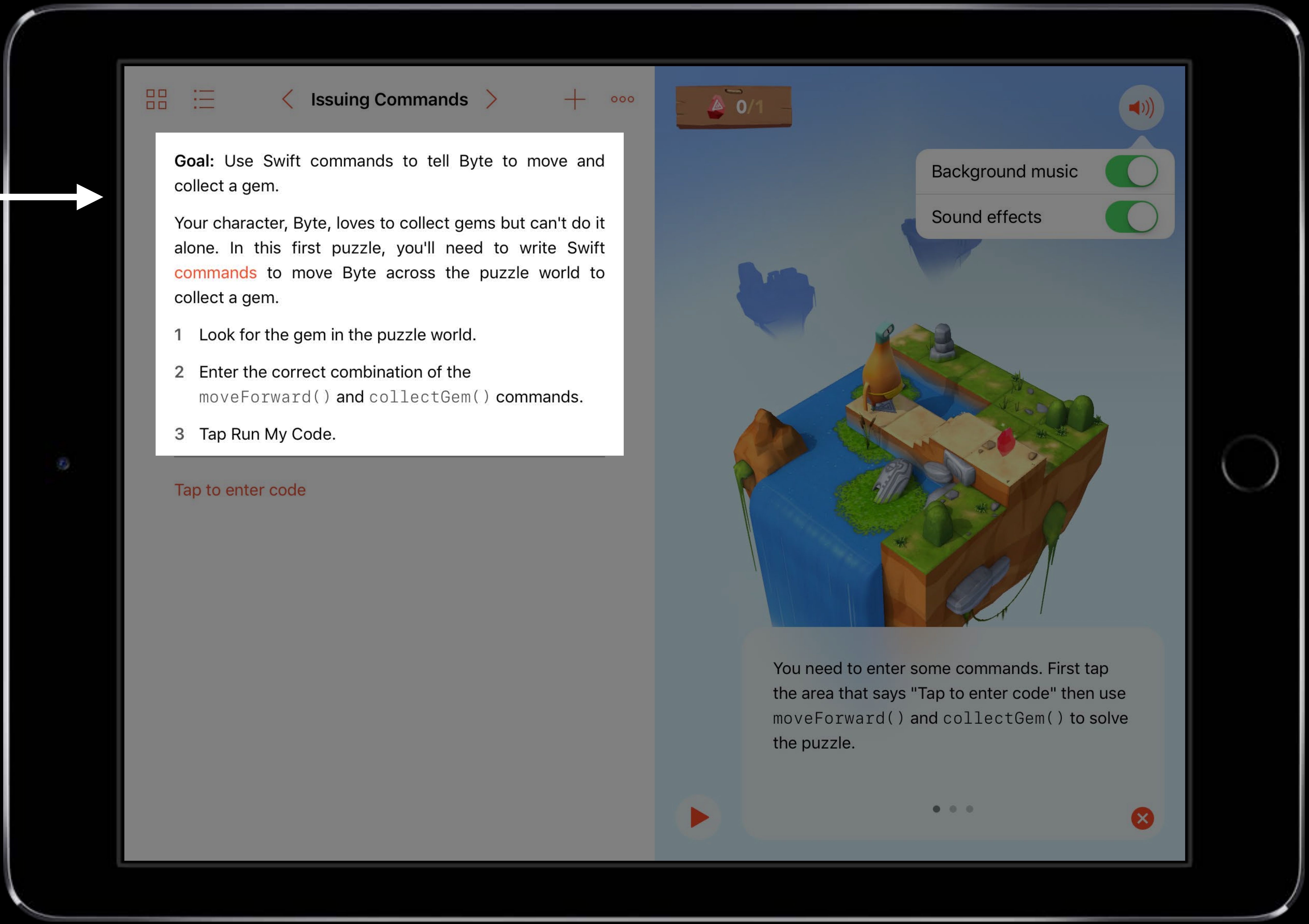
Background music: on

Sound effects: on

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.

Live view

Prose



Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

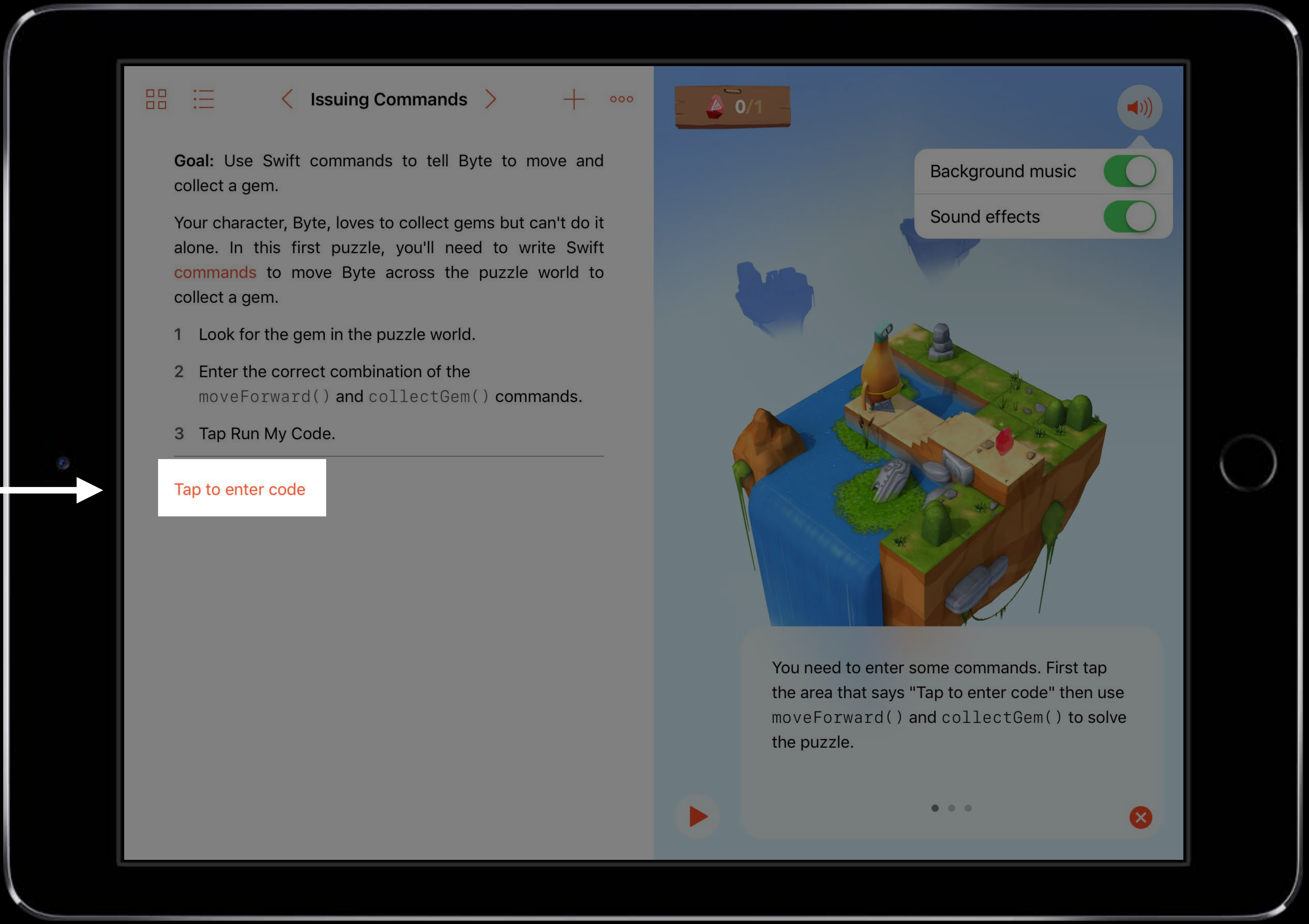
1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.

Editable code placeholder



Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

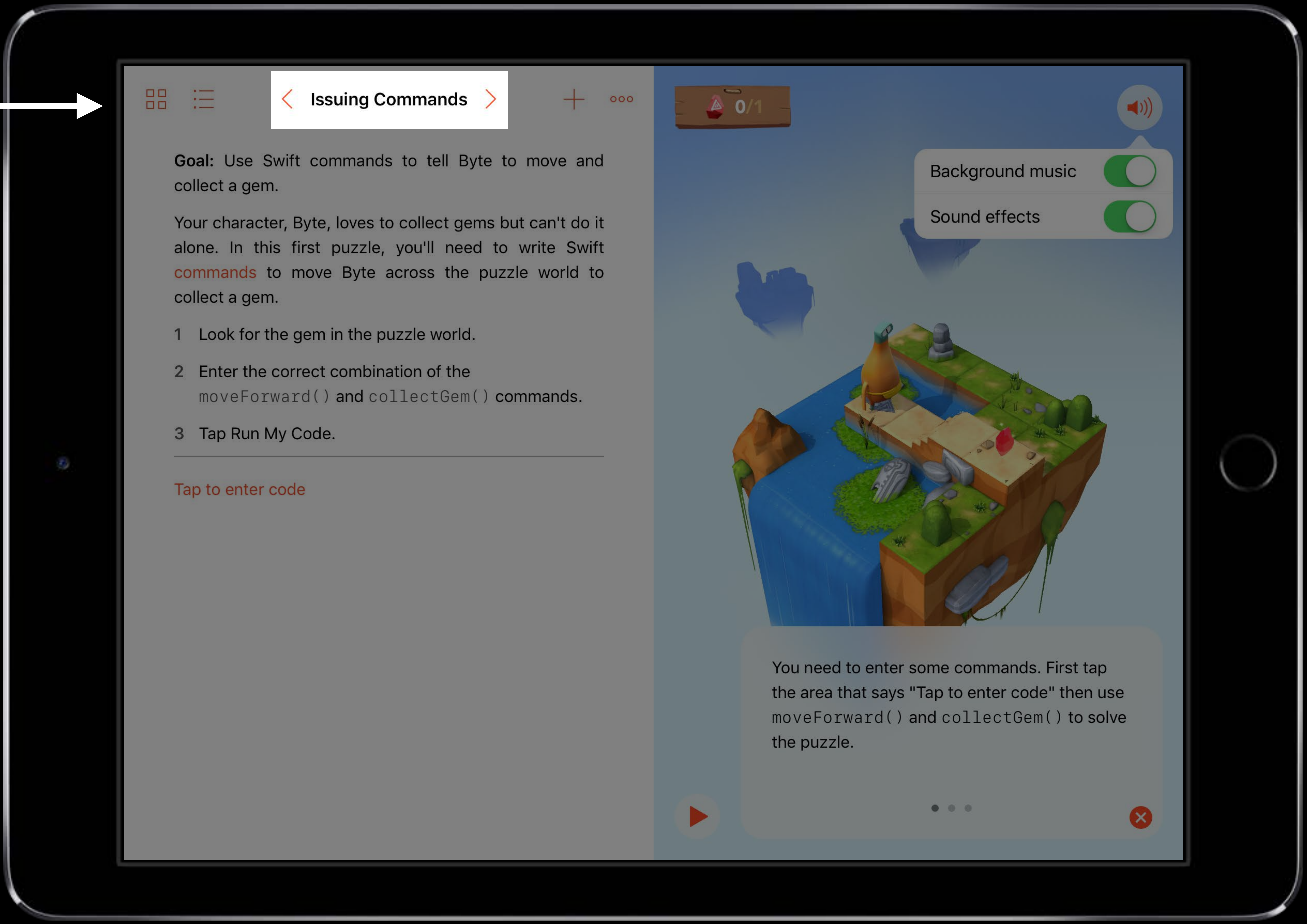
2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

Tap to enter code

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.

Page title



Issuing Commands

Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

Tap to enter code

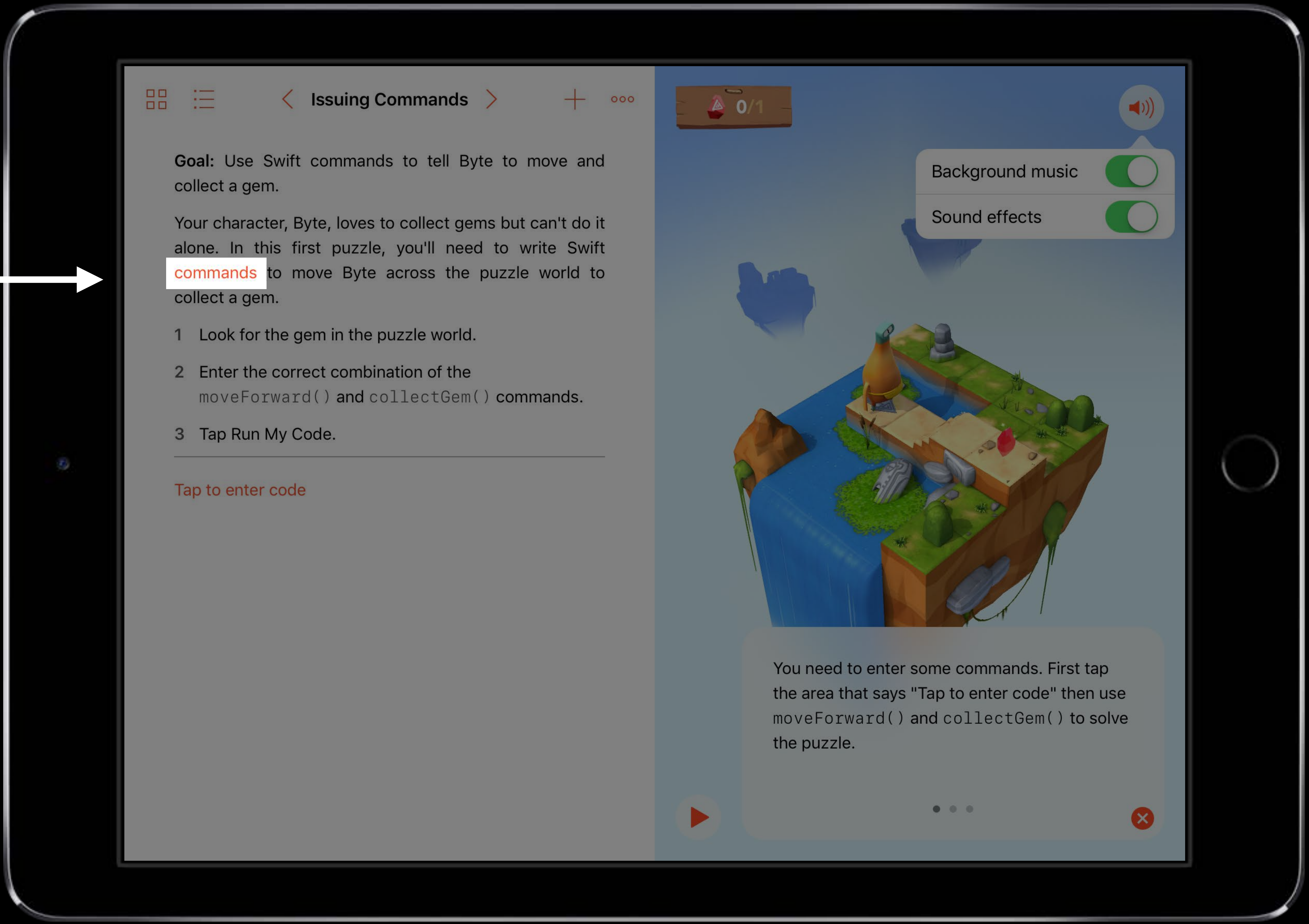
0/1

Background music

Sound effects

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.

Glossary term



Goal: Use Swift commands to tell Byte to move and collect a gem.

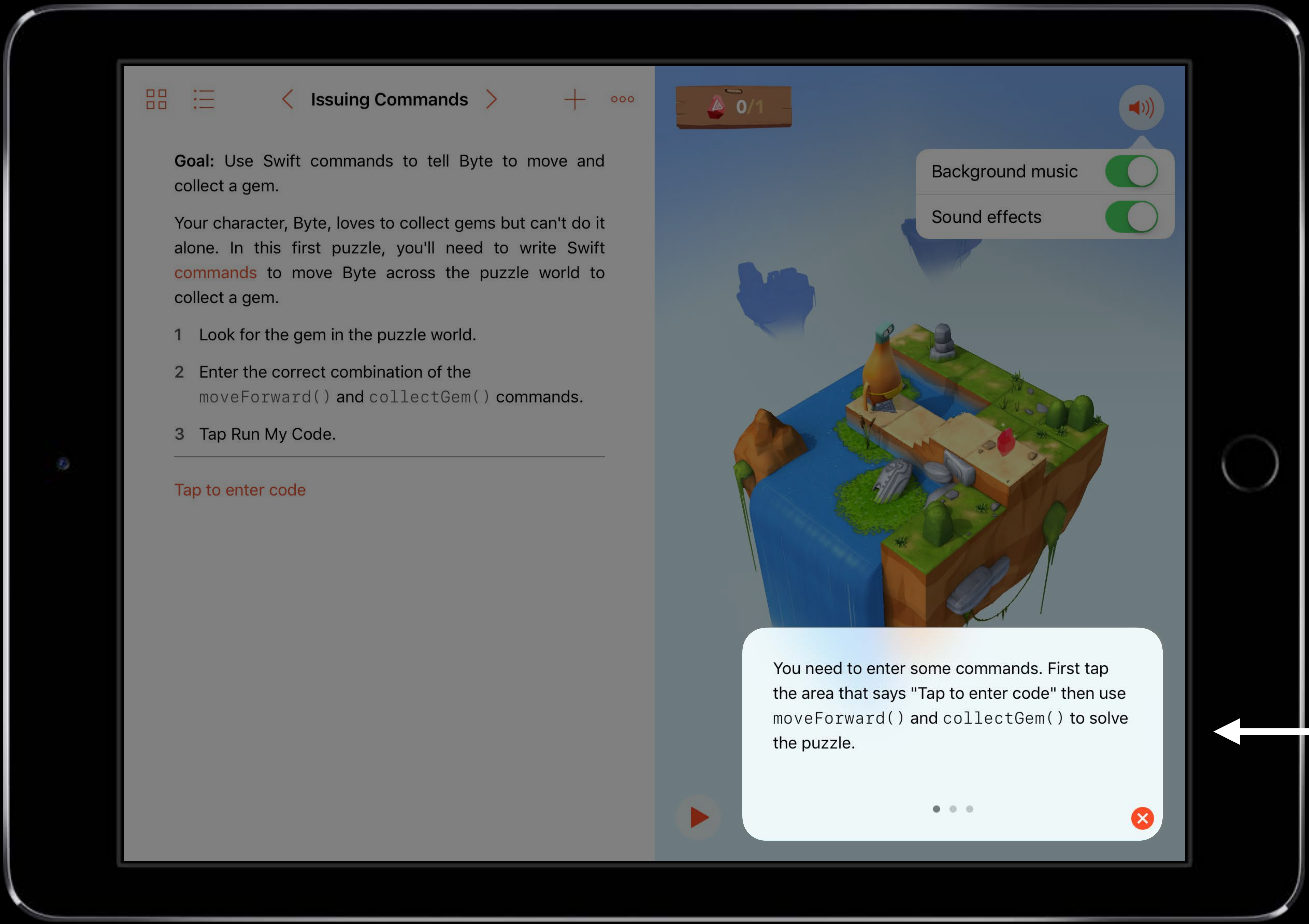
Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.



Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

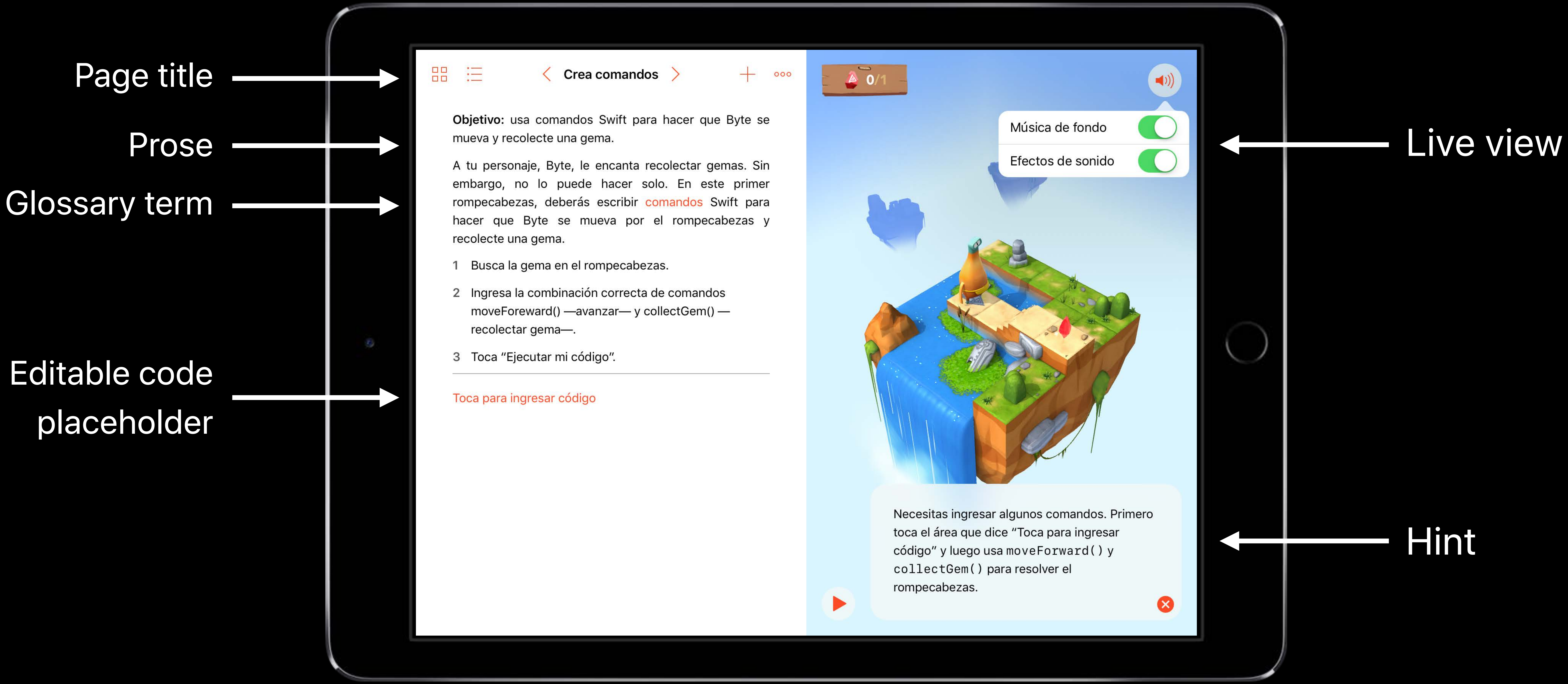
3 Tap Run My Code.

Background music: on

Sound effects: on

You need to enter some commands. First tap the area that says "Tap to enter code" then use `moveForward()` and `collectGem()` to solve the puzzle.

Hint



Page title

Prose

Glossary term

Editable code placeholder

Live view

Hint

Crea comandos

Objetivo: usa comandos Swift para hacer que Byte se mueva y recolecte una gema.

A tu personaje, Byte, le encanta recolectar gemas. Sin embargo, no lo puede hacer solo. En este primer rompecabezas, deberás escribir comandos Swift para hacer que Byte se mueva por el rompecabezas y recolecte una gema.

1 Busca la gema en el rompecabezas.

2 Ingresa la combinación correcta de comandos moveForeward() —avanzar— y collectGem() —recolectar gema—.

3 Toca “Ejecutar mi código”.

Toca para ingresar código

Música de fondo ●

Efectos de sonido ●

Necesitas ingresar algunos comandos. Primero toca el área que dice “Toca para ingresar código” y luego usa `moveForward()` y `collectGem()` para resolver el rompecabezas.

Files Types to Localize

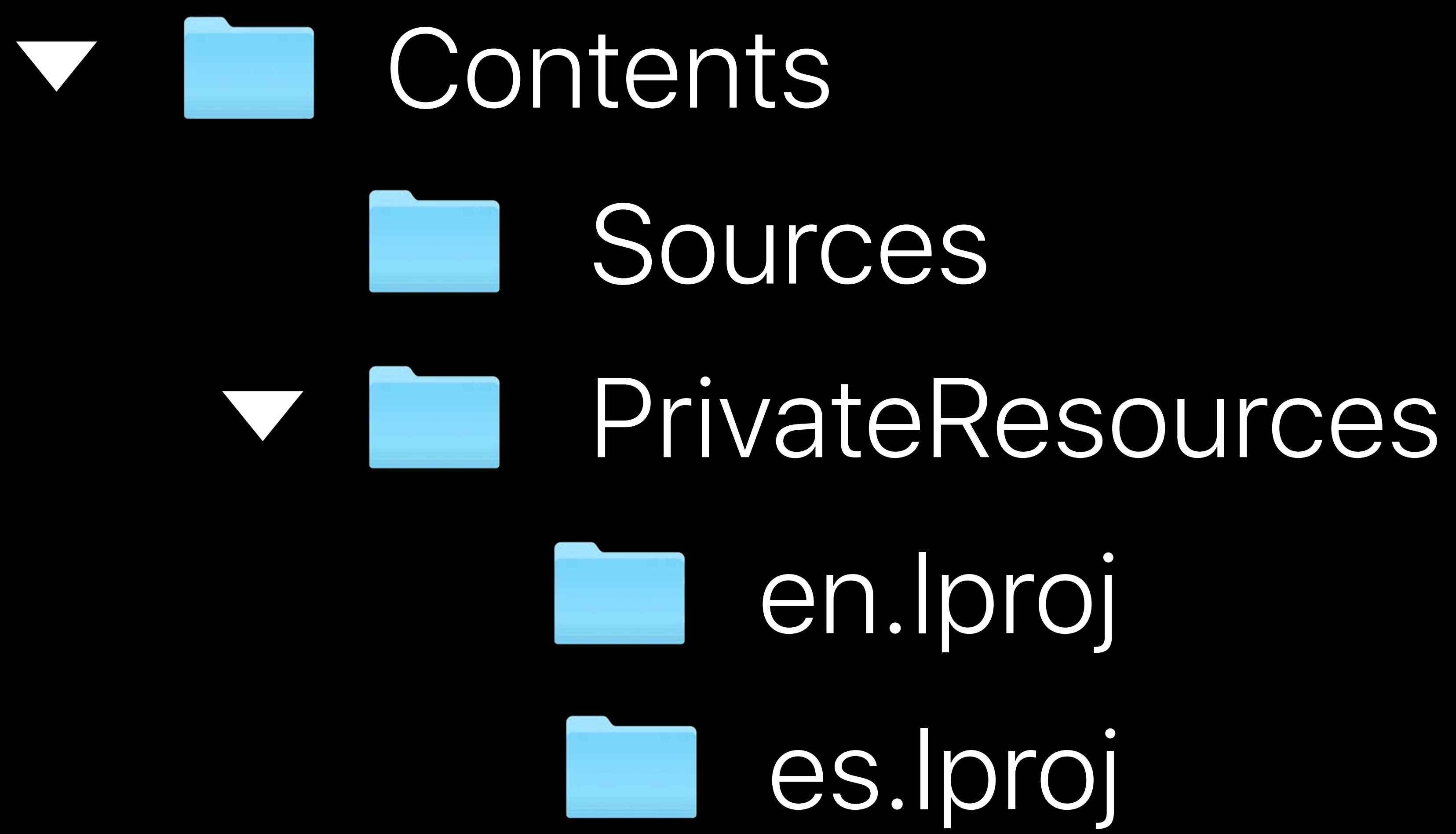
*.strings

*.stringsdict

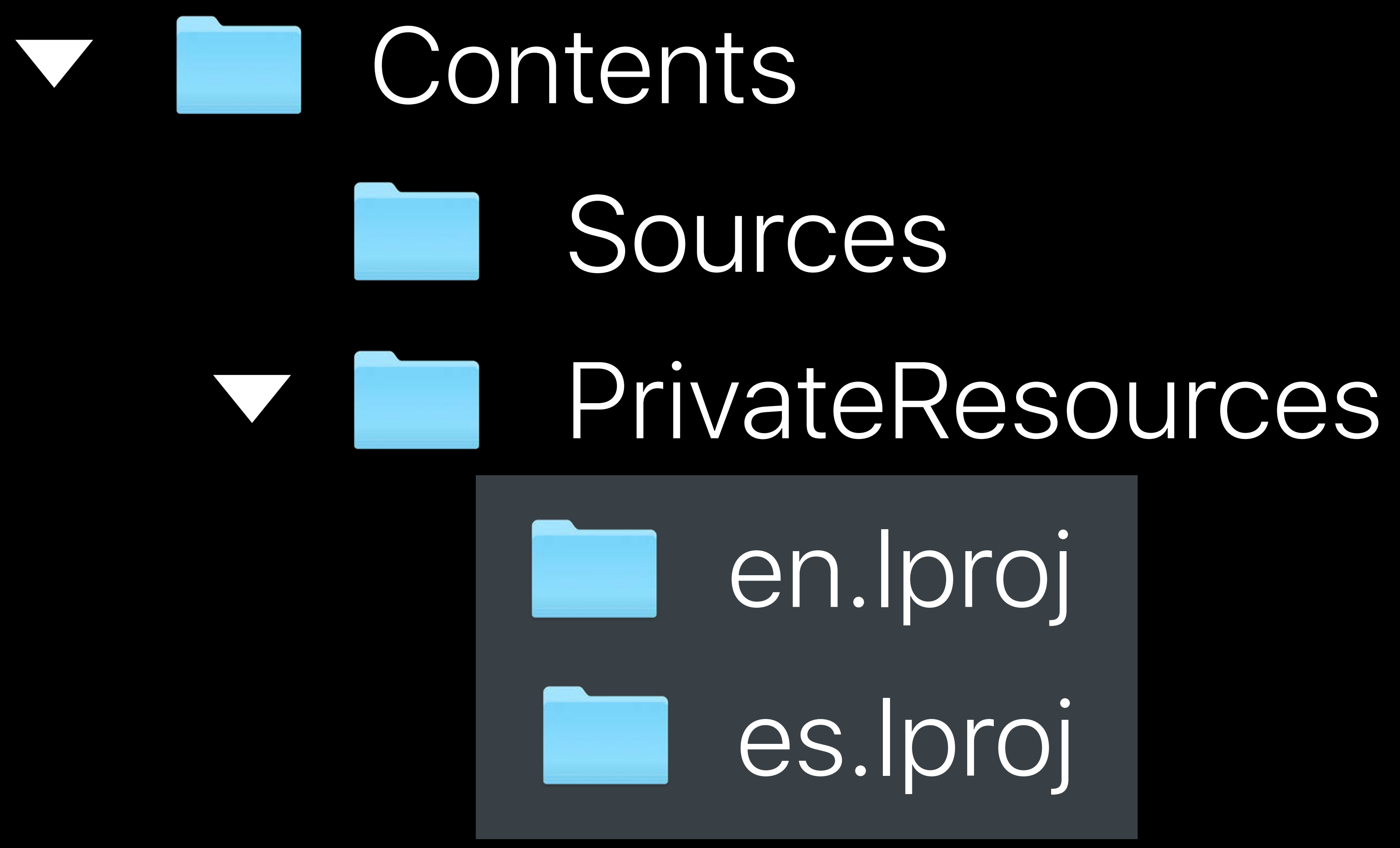
*.plist

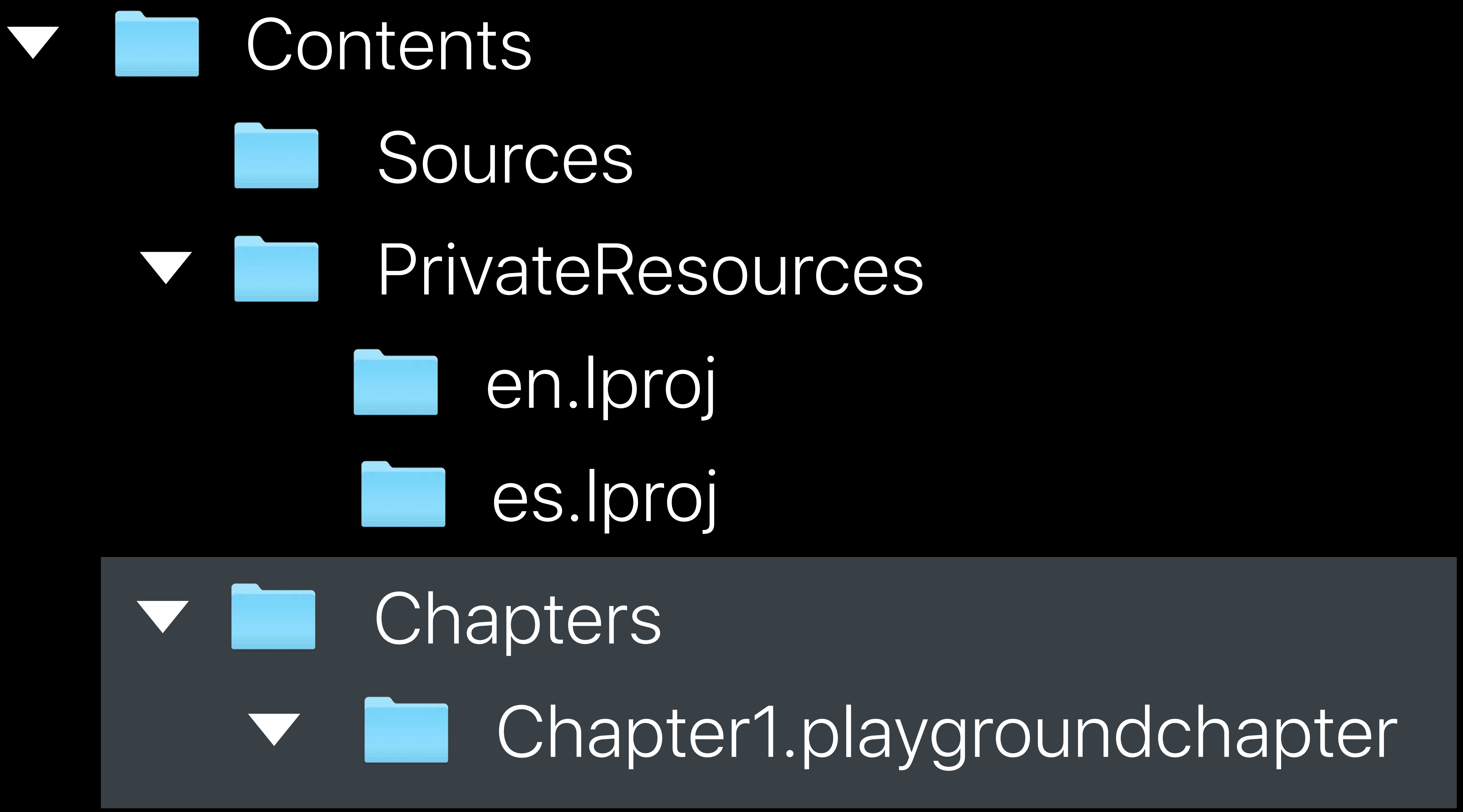
*.html

Localized assets

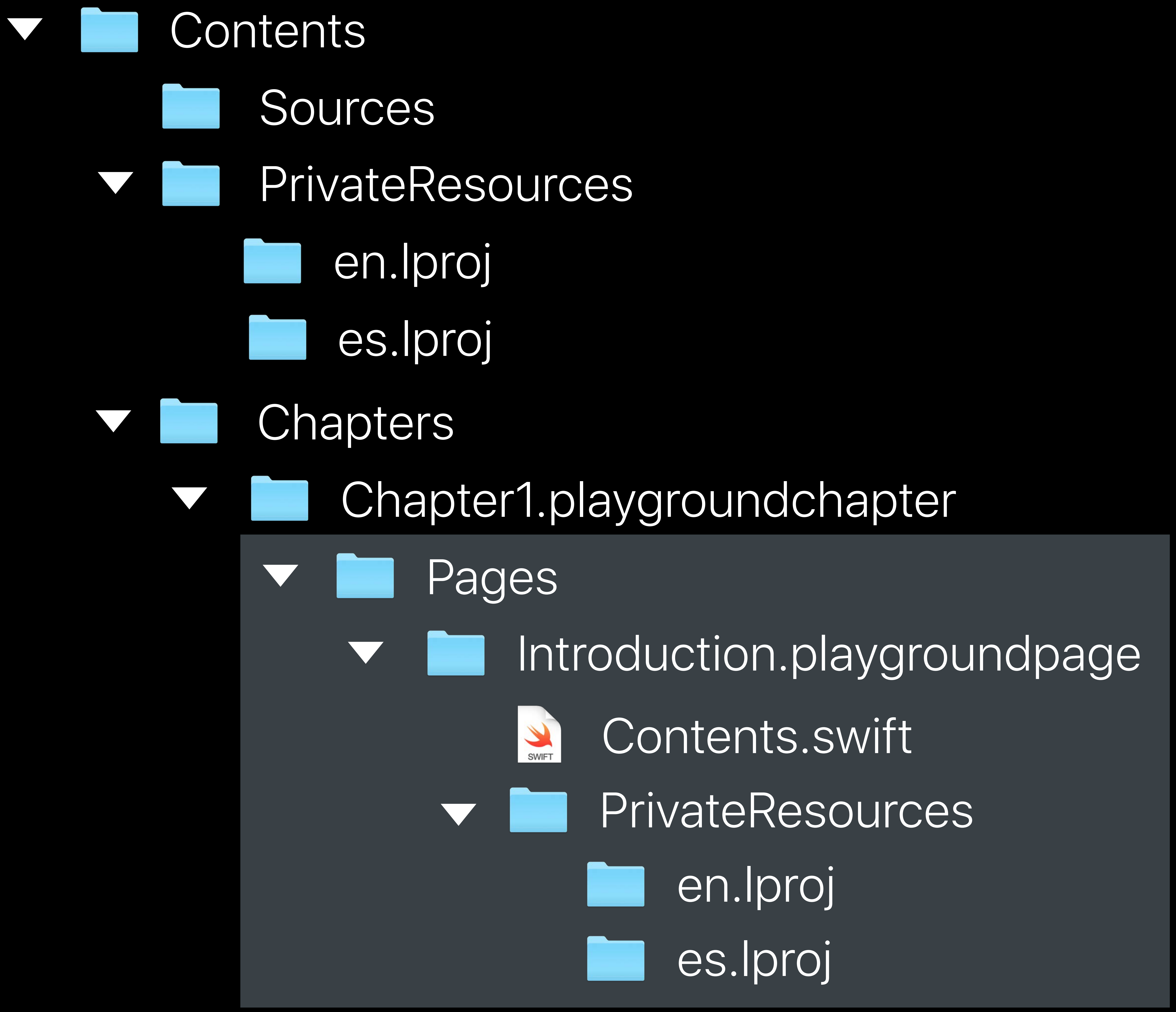


- Contents
  - Sources
  - PrivateResources
    - en.lproj
    - es.lproj

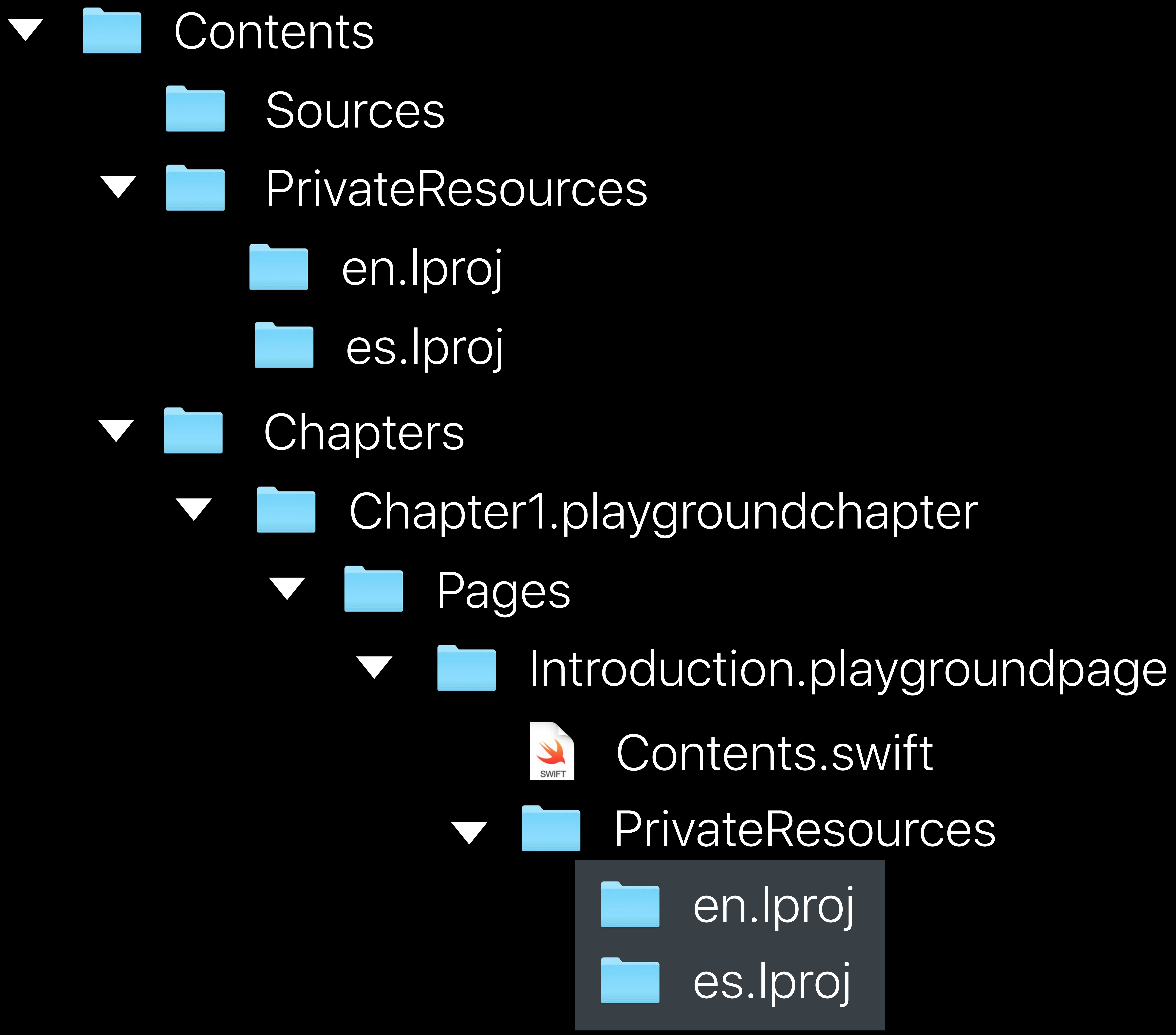

Contents
Sources
PrivateResources
en.lproj
es.lproj



- Contents
  - Sources
  - PrivateResources
    - en.lproj
    - es.lproj
  - Chapters
    - Chapter1.playgroundchapter



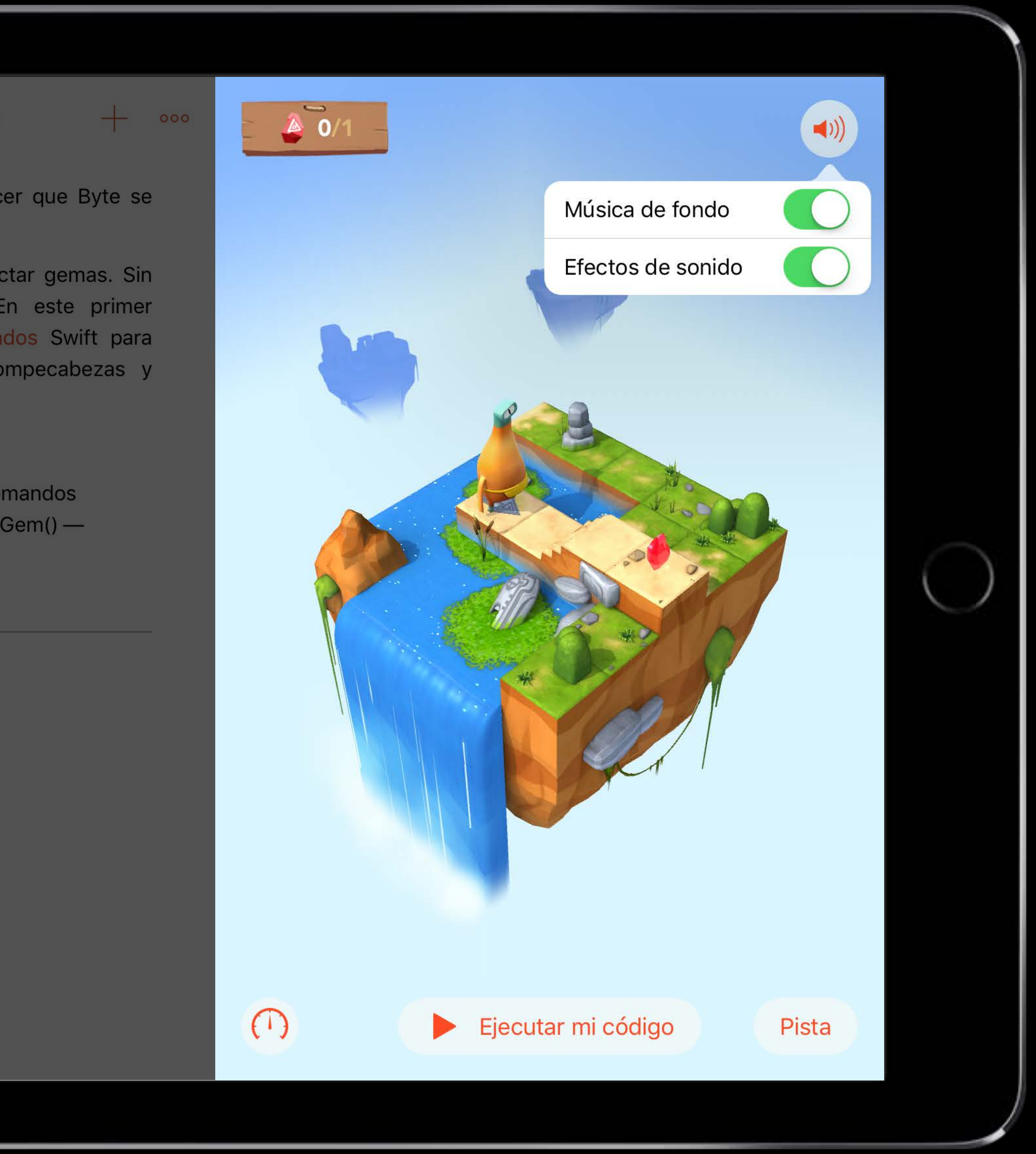
```
Contents
    Sources
    PrivateResources
        en.lproj
        es.lproj
    Chapters
        Chapter1.playgroundchapter
            Pages
                Introduction.playgroundpage
                    Contents.swift
                    PrivateResources
                        en.lproj
                        es.lproj
```



```
Contents
    Sources
    PrivateResources
        en.lproj
        es.lproj
    Chapters
        Chapter1.playgroundchapter
            Pages
                Introduction.playgroundpage
                    Contents.swift
                    PrivateResources
                        en.lproj
                        es.lproj
```

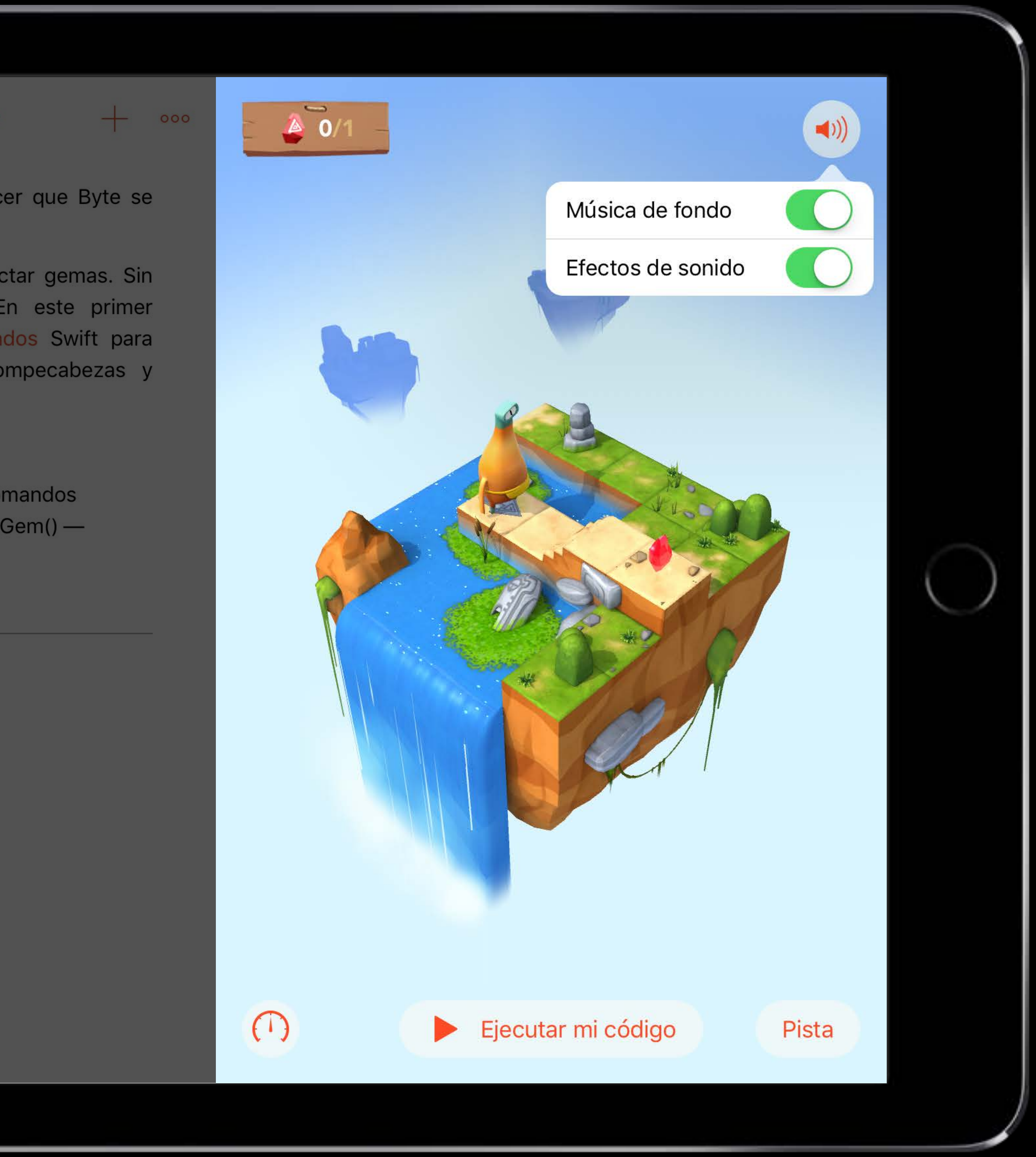
Strings in Live View

*.strings files



Strings in Live View

*.strings files



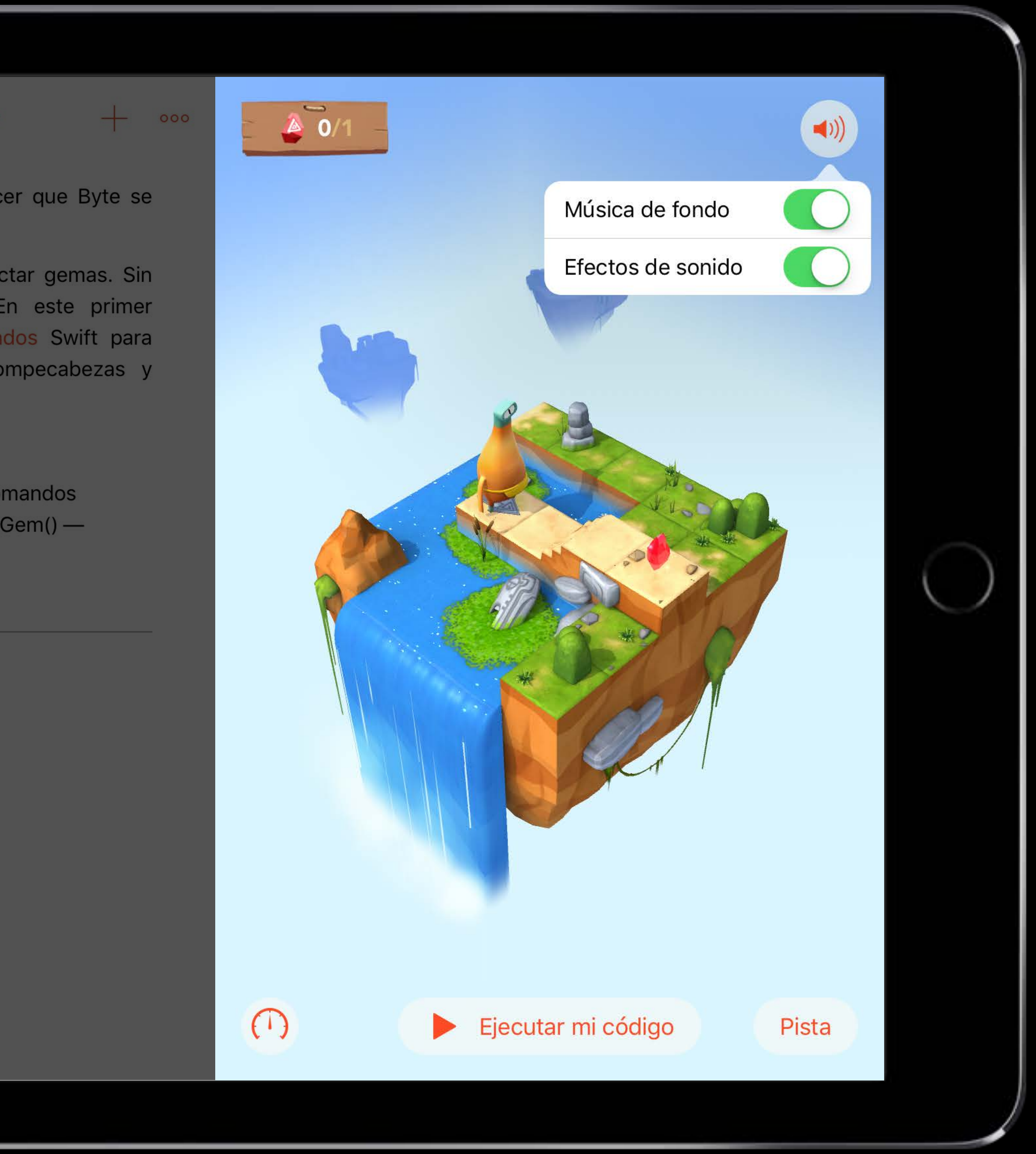
Música de fondo ●

Efectos de sonido ●

Use `NSLocalizedString()`

Strings in Live View

*.strings files



cer que Byte se

ctar gemas. Sin
En este primer
dos Swift para
mpecabezas y

mandos
Gem() —

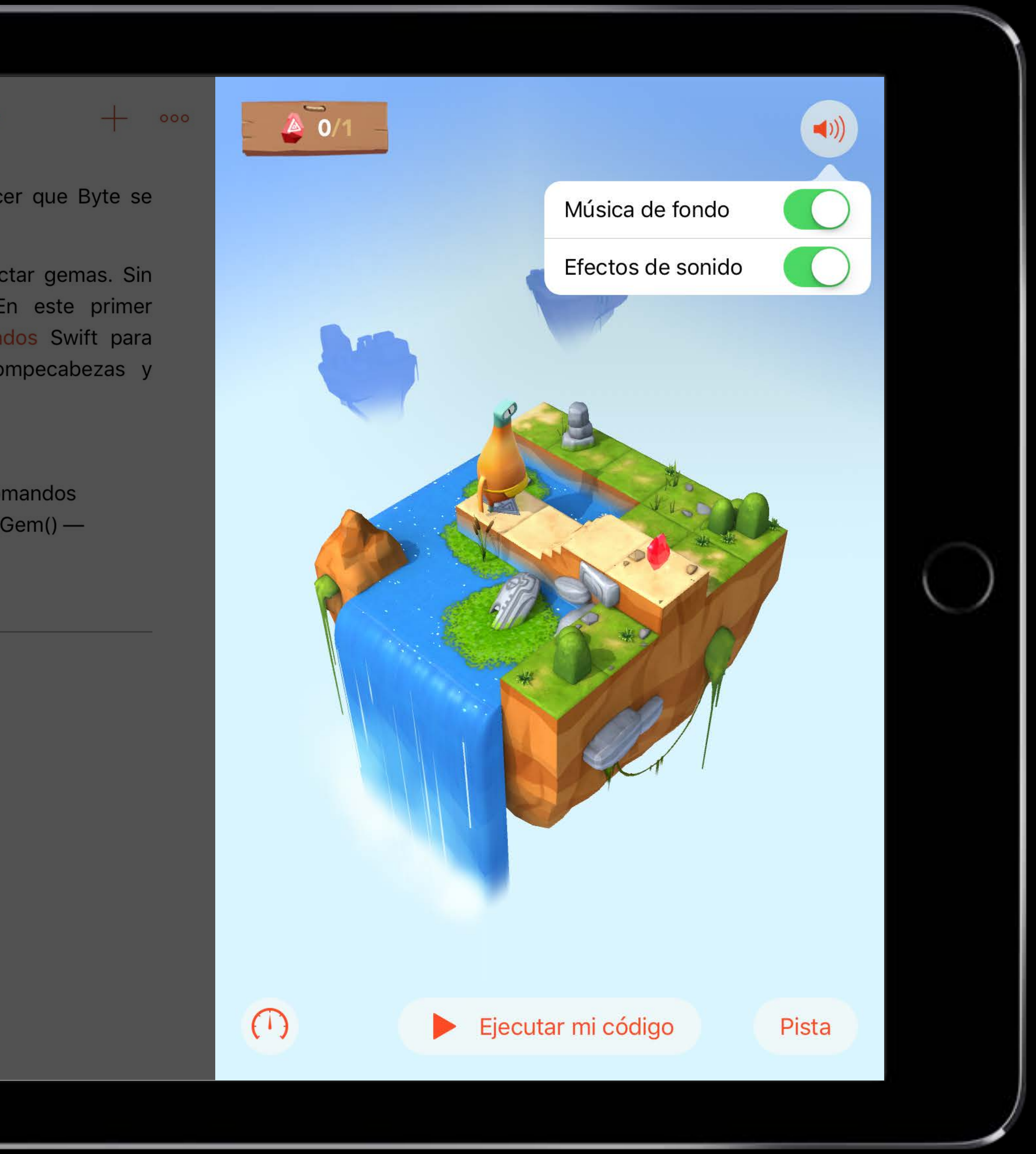
Música de fondo
Efectos de sonido

Use `NSLocalizedString()`

Extract with `genstrings`

Strings in Live View

*.strings files



Música de fondo ●

Efectos de sonido ●

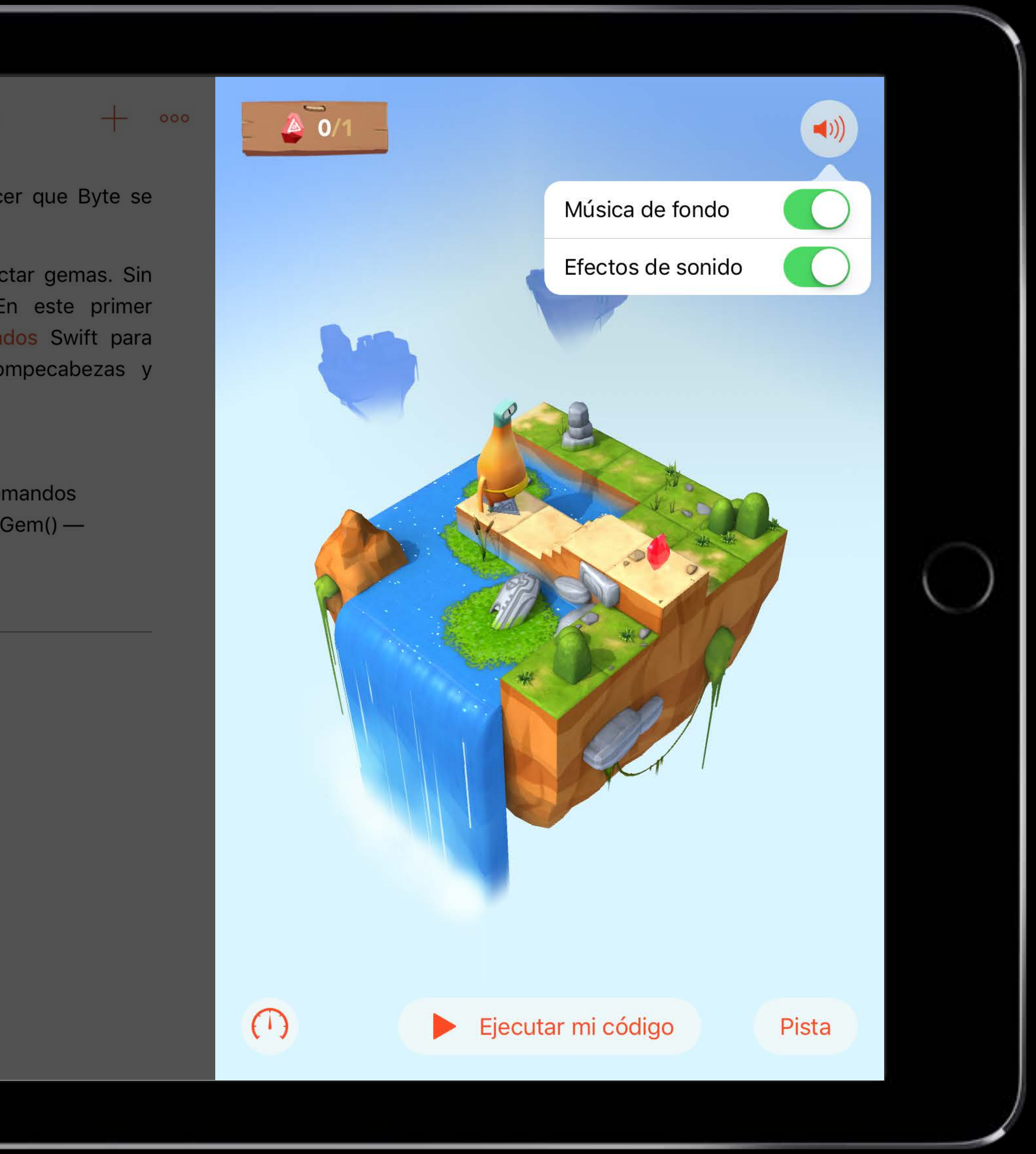
Use `NSLocalizedString()`

Extract with `genstrings`

More information
http://developer.apple.com/go/?id=string-resources

Strings in Live View

*.strings files



Música de fondo ●

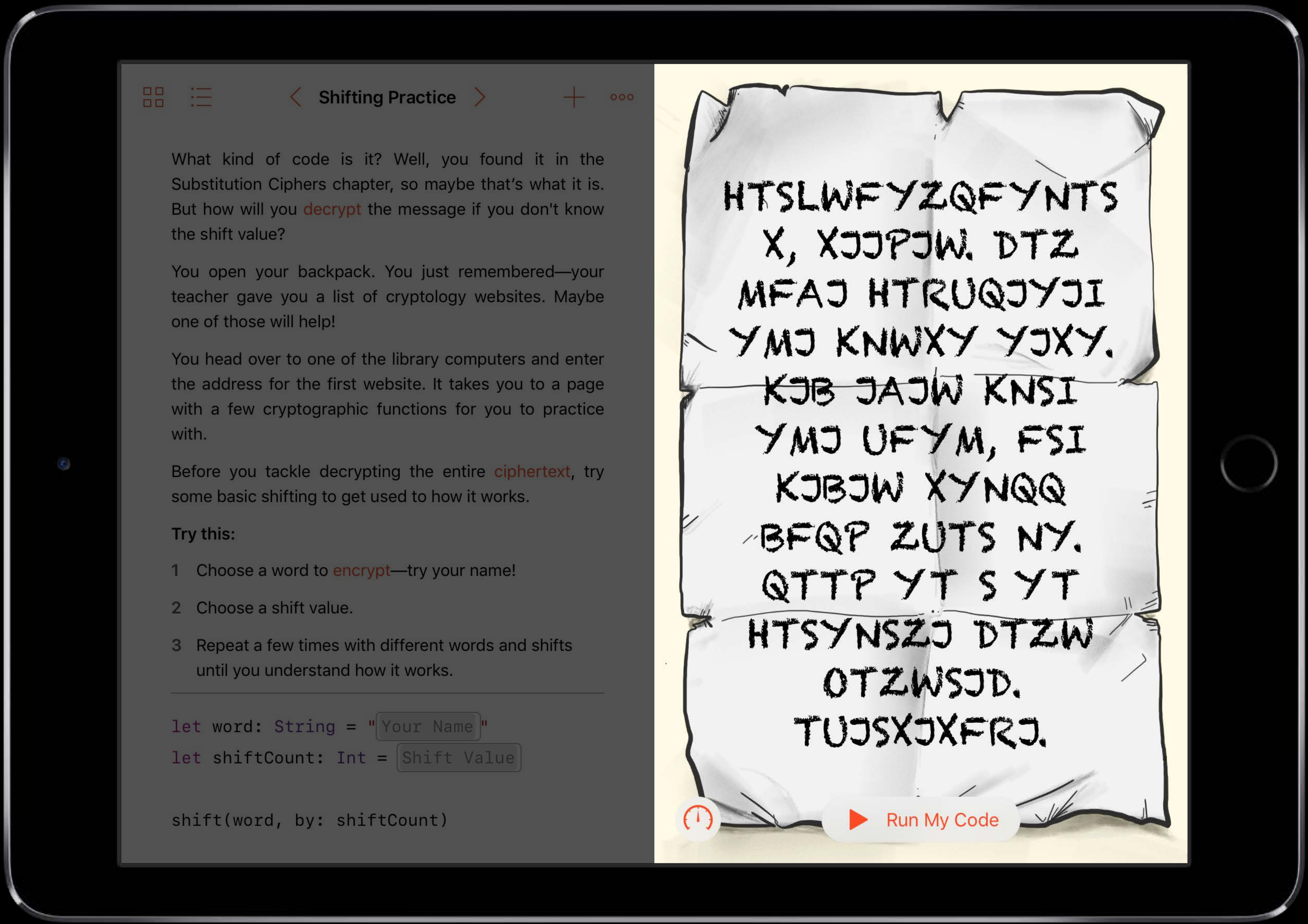
Efectos de sonido ●

Use `NSLocalizedString()`

Extract with `genstrings`

More information

http://developer.apple.com/go/?id=string-resources



What kind of code is it? Well, you found it in the Substitution Ciphers chapter, so maybe that's what it is. But how will you decrypt the message if you don't know the shift value?

You open your backpack. You just remembered—your teacher gave you a list of cryptology websites. Maybe one of those will help!

You head over to one of the library computers and enter the address for the first website. It takes you to a page with a few cryptographic functions for you to practice with.

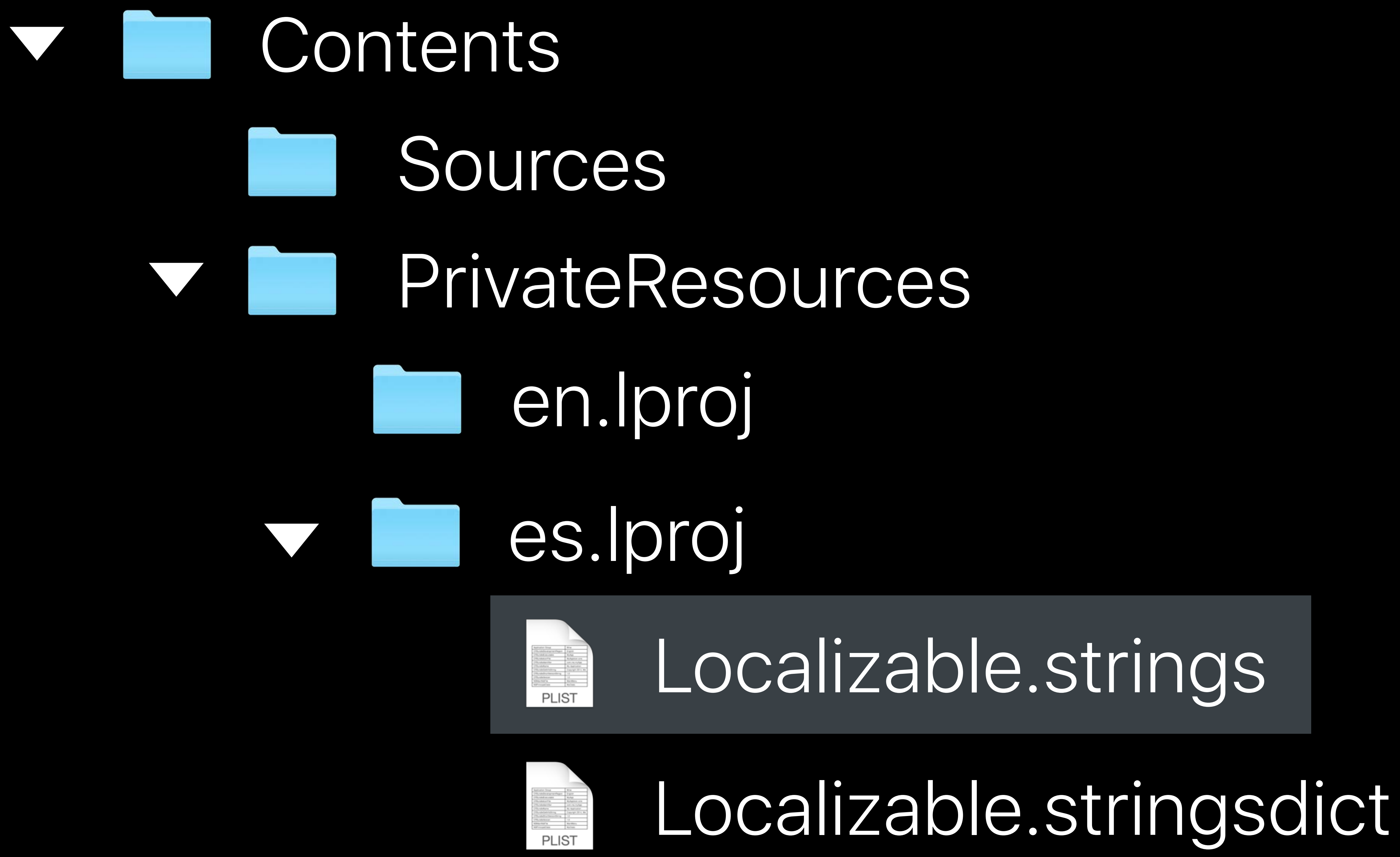
Before you tackle decrypting the entire ciphertext, try some basic shifting to get used to how it works.

Try this:

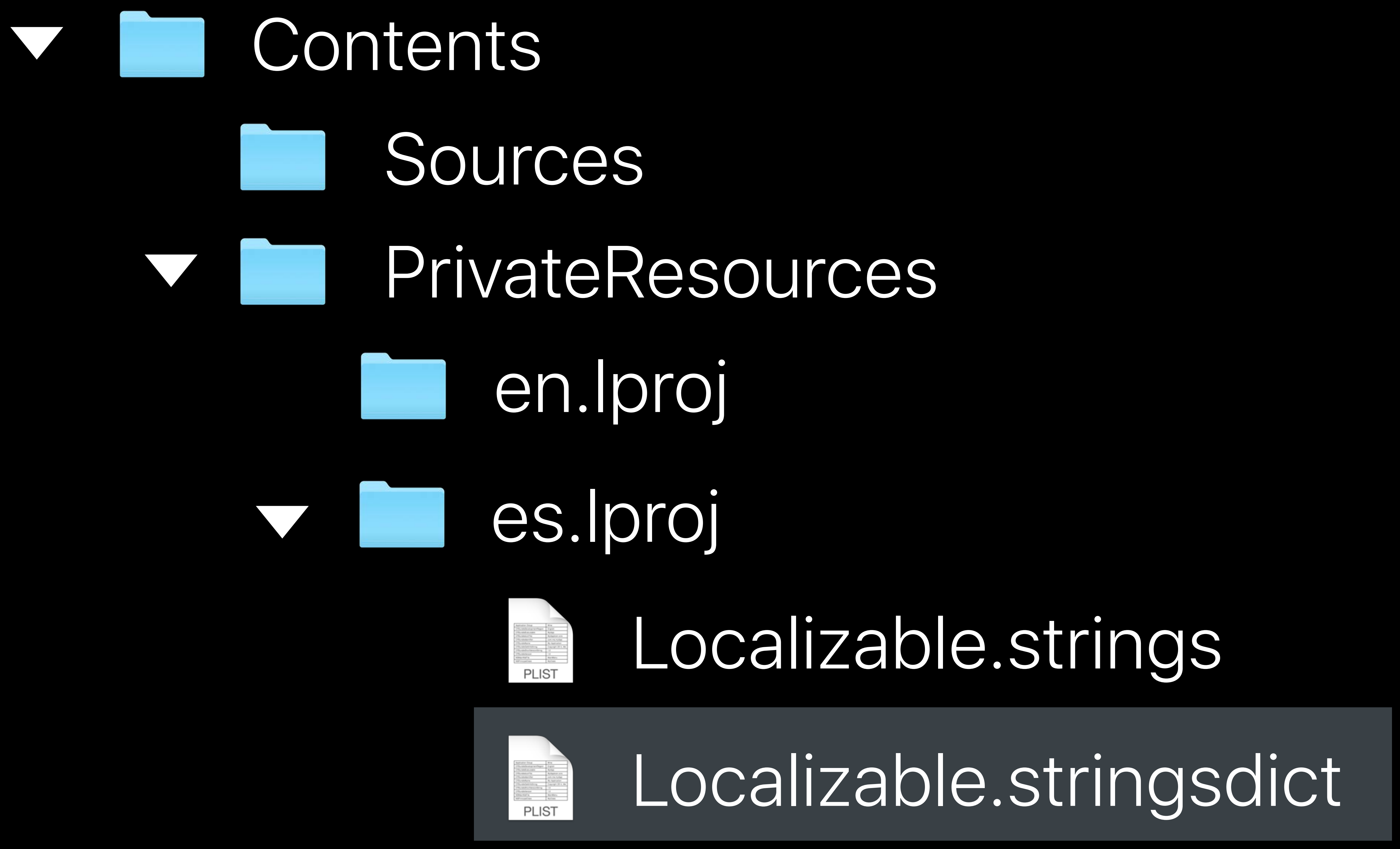
1. Choose a word to encrypt—try your name!
2. Choose a shift value.
3. Repeat a few times with different words and shifts until you understand how it works.

```
let word: String = "Your Name"
let shiftCount: Int = Shift Value

shift(word, by: shiftCount)
```

HTSLWFYZQFYNTS
X, XJJPJW. DTZ
MFAJ HTRUQJYJI
YMJ KNWXY YJXY.
KJB JAJW KNSI
YMJ UFYM, FSI
KJBJW XYNQQ
BFQP ZUTS NY.
QTTP YT S YT
HTSYNSZJ DTZW
OTZWSJD.
TUJSXJXFRJ.



```
Contents
    Sources
    PrivateResources
        en.lproj
        es.lproj
            Localizable.strings
            Localizable.stringsdict
```

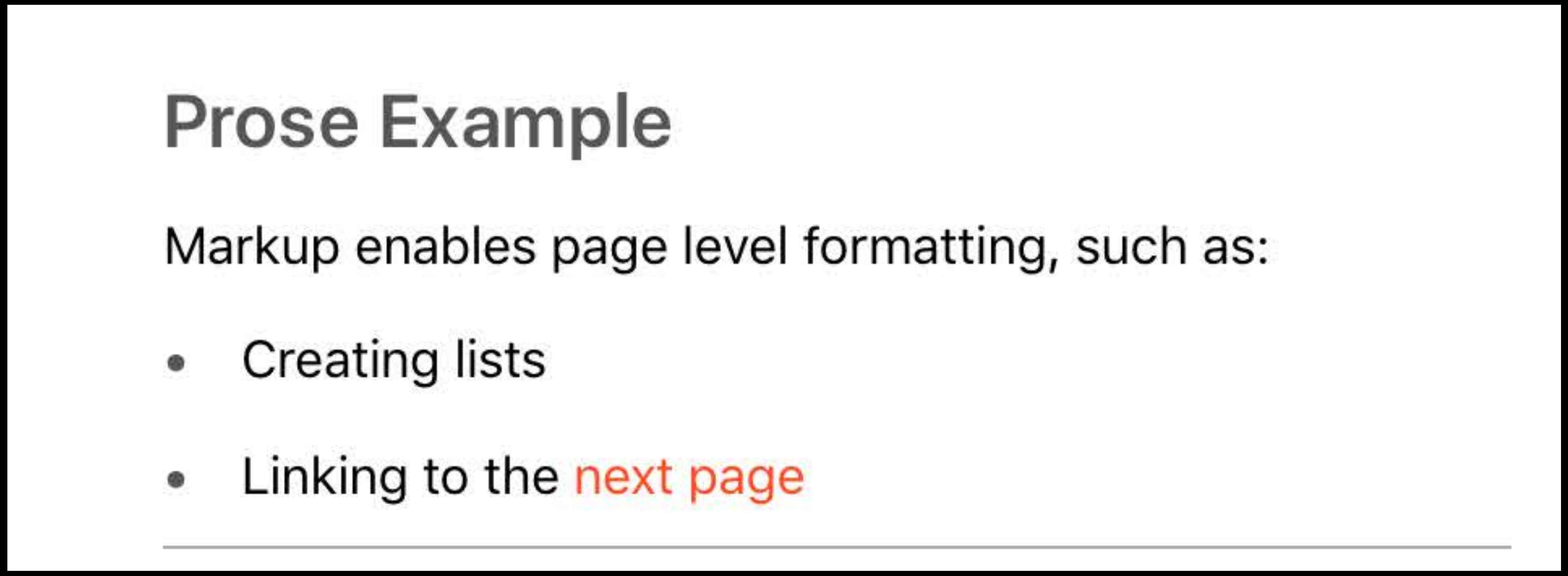


```
Contents
    Sources
    PrivateResources
        en.lproj
        es.lproj
            Localizable.strings
            Localizable.stringsdict
```

Prose

*.strings files

Rendered prose



Prose Example

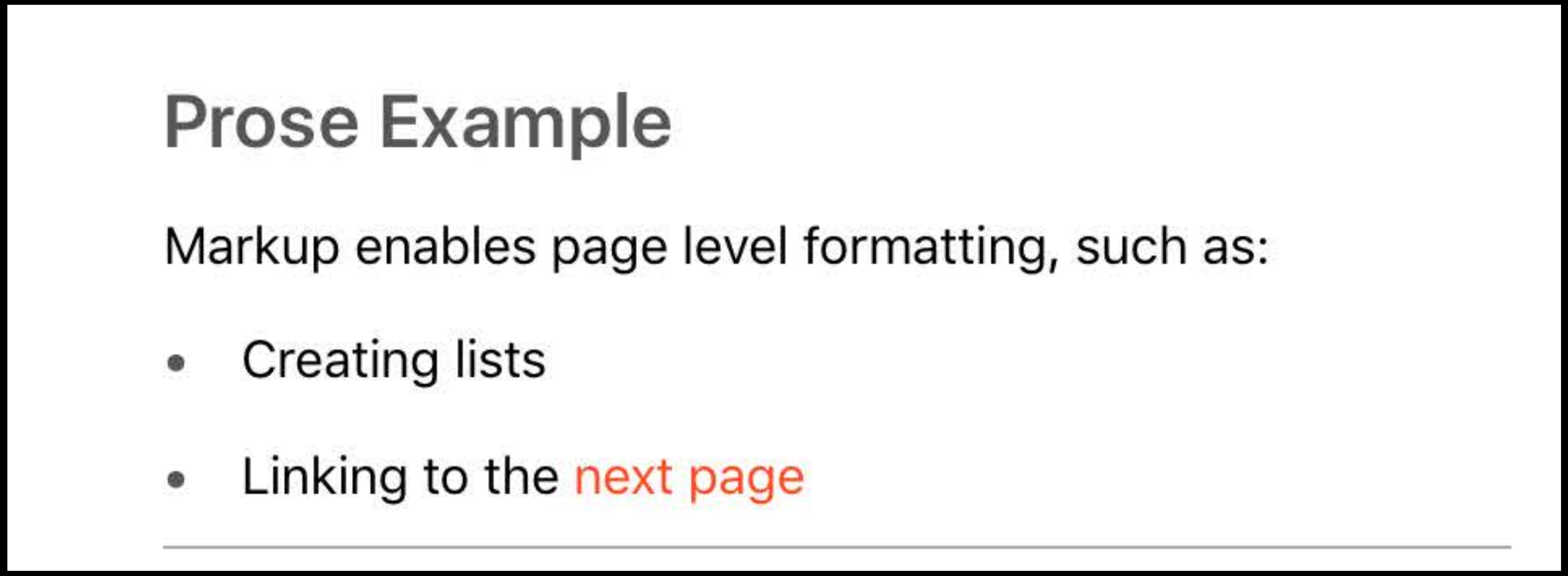
Markup enables page level formatting, such as:

- Creating lists
- Linking to the next page

Prose

*.strings files

Rendered prose



Prose Example

Markup enables page level formatting, such as:

- Creating lists
- Linking to the next page

Markup

```
/*:
 # Prose Example
 Markup enables page level formatting, such as:
  - Creating lists
  - Linking to the [next page](@next)
*/
```

Prose

*.strings files

Contents.swift

```
/*:

*/
```

Prose

*.strings files

Contents.swift

```
/*:#localized(key: "FirstProseBlock")

*/
```

Prose

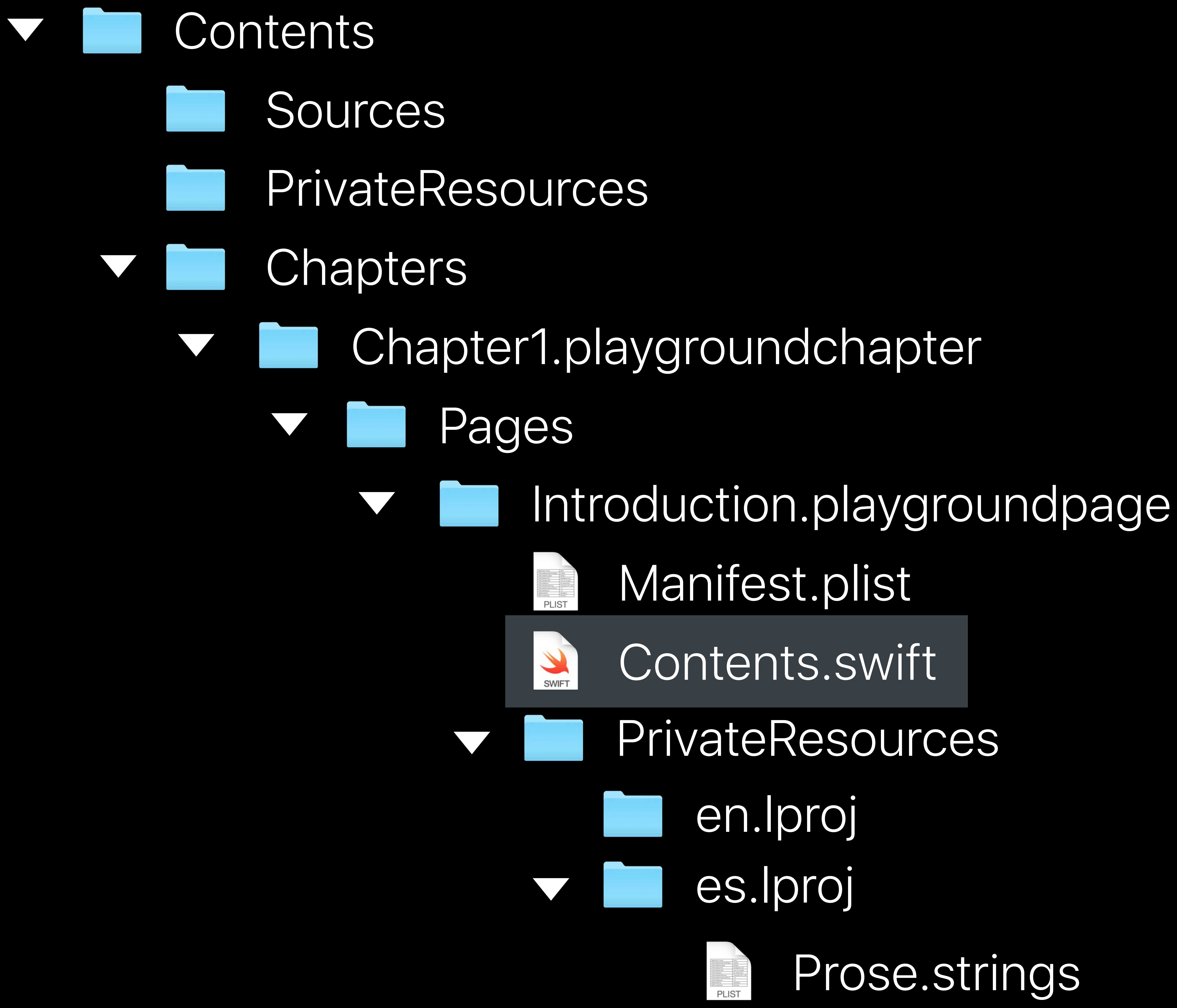
*.strings files

Contents.swift

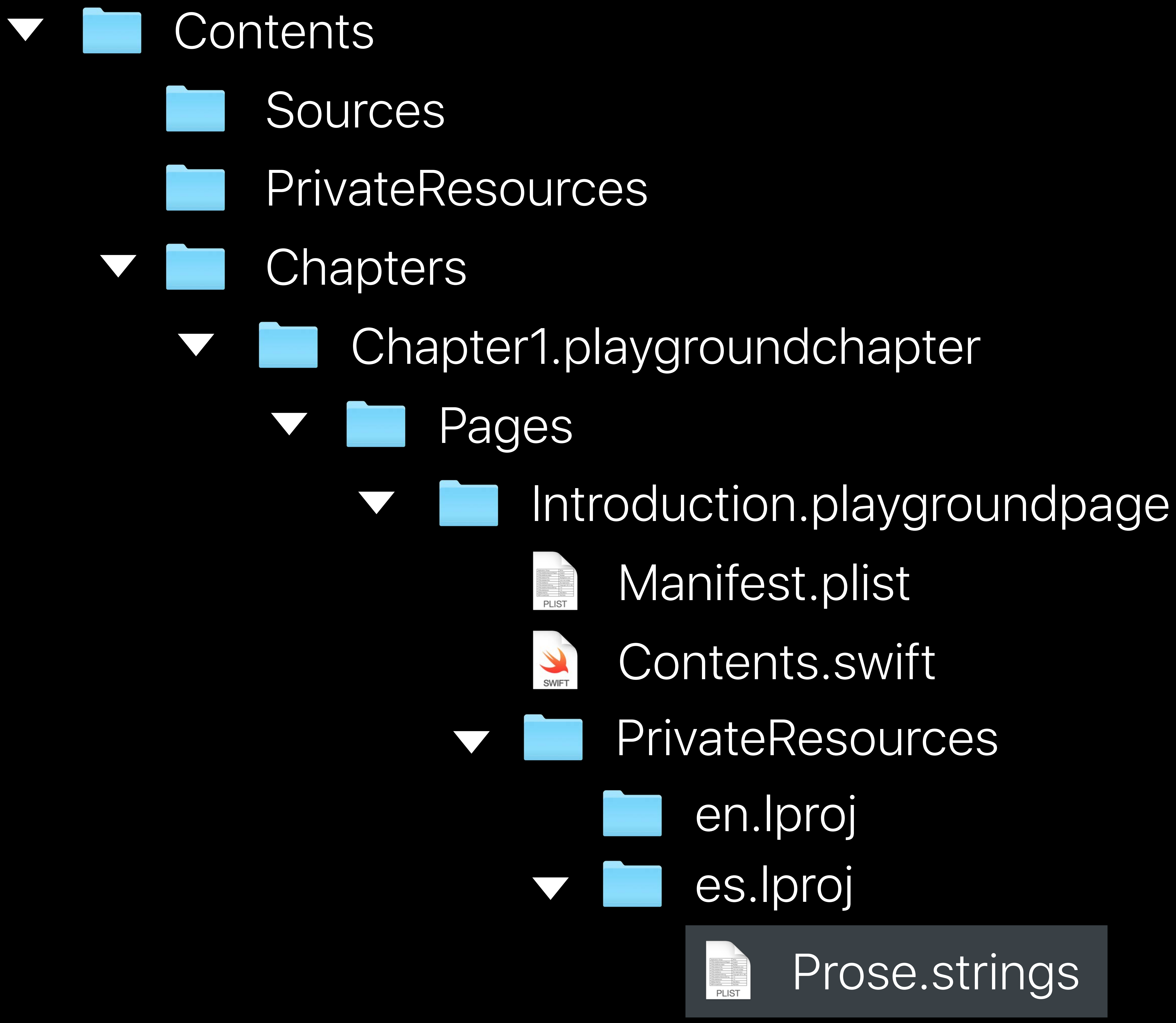
```
/*:#localized(key: "FirstProseBlock")

*/
```

en.lproj/Prose.strings

```
"FirstProseBlock" = "Write your *localized prose* here in Markup";
```



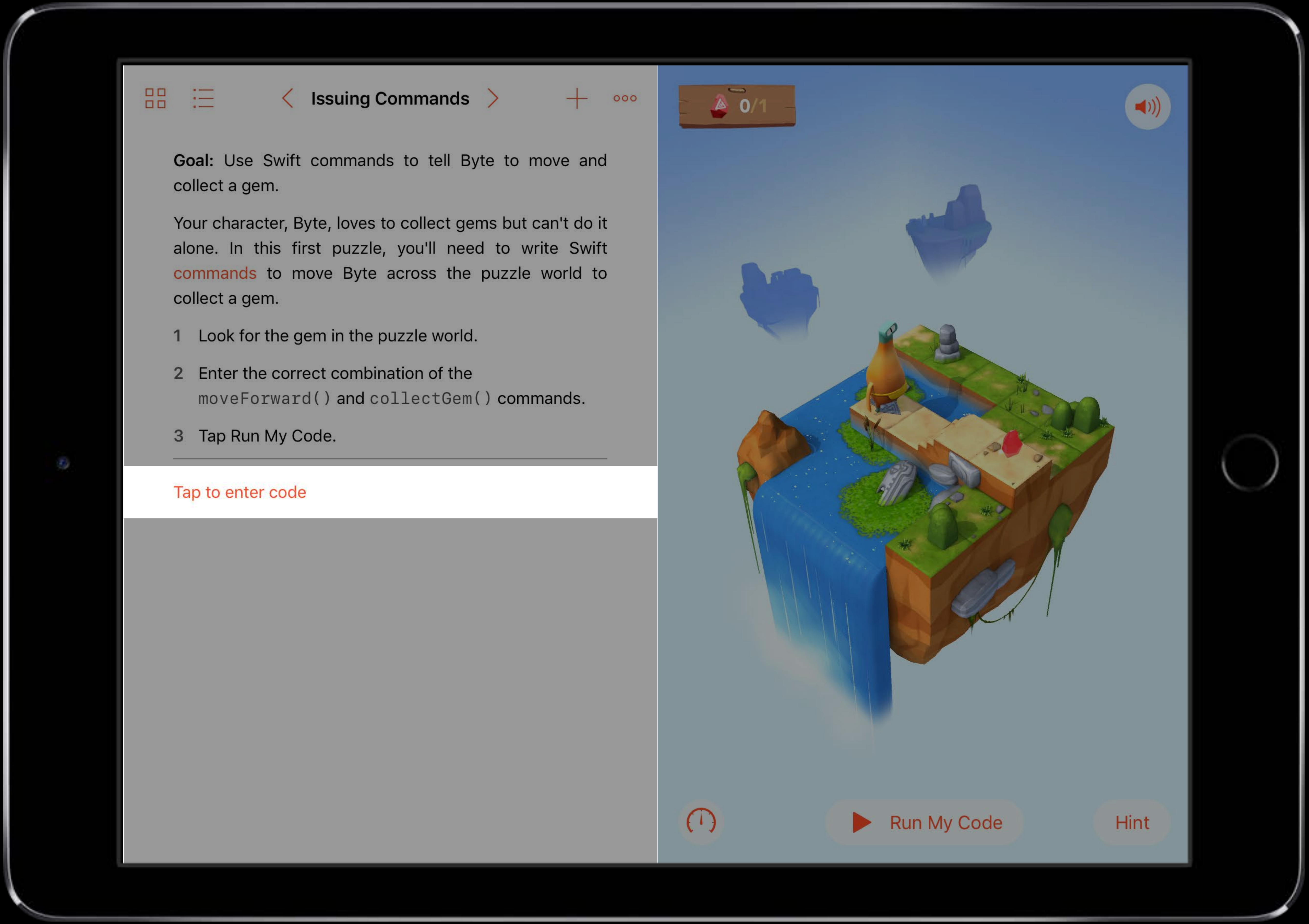
- Contents
  - Sources
  - PrivateResources
  - Chapters
    - Chapter1.playgroundchapter
      - Pages
        - Introduction.playgroundpage
          - Manifest.plist
          - Contents.swift
          - PrivateResources
            - en.lproj
            - es.lproj
              - Prose.strings



- Contents
  - Sources
  - PrivateResources
  - Chapters
    - Chapter1.playgroundchapter
      - Pages
        - Introduction.playgroundpage
          - Manifest.plist
          - Contents.swift
          - PrivateResources
            - en.lproj
            - es.lproj
              - Prose.strings

Editable Text Field Placeholders

*.strings files



Goal: Use Swift commands to tell Byte to move and collect a gem.

Your character, Byte, loves to collect gems but can't do it alone. In this first puzzle, you'll need to write Swift commands to move Byte across the puzzle world to collect a gem.

1 Look for the gem in the puzzle world.

2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.

3 Tap Run My Code.

Tap to enter code

Editable Text Field Placeholders

*.strings files

Contents.swift

```
//#-editable-code Tap to enter code
```

Editable Text Field Placeholders

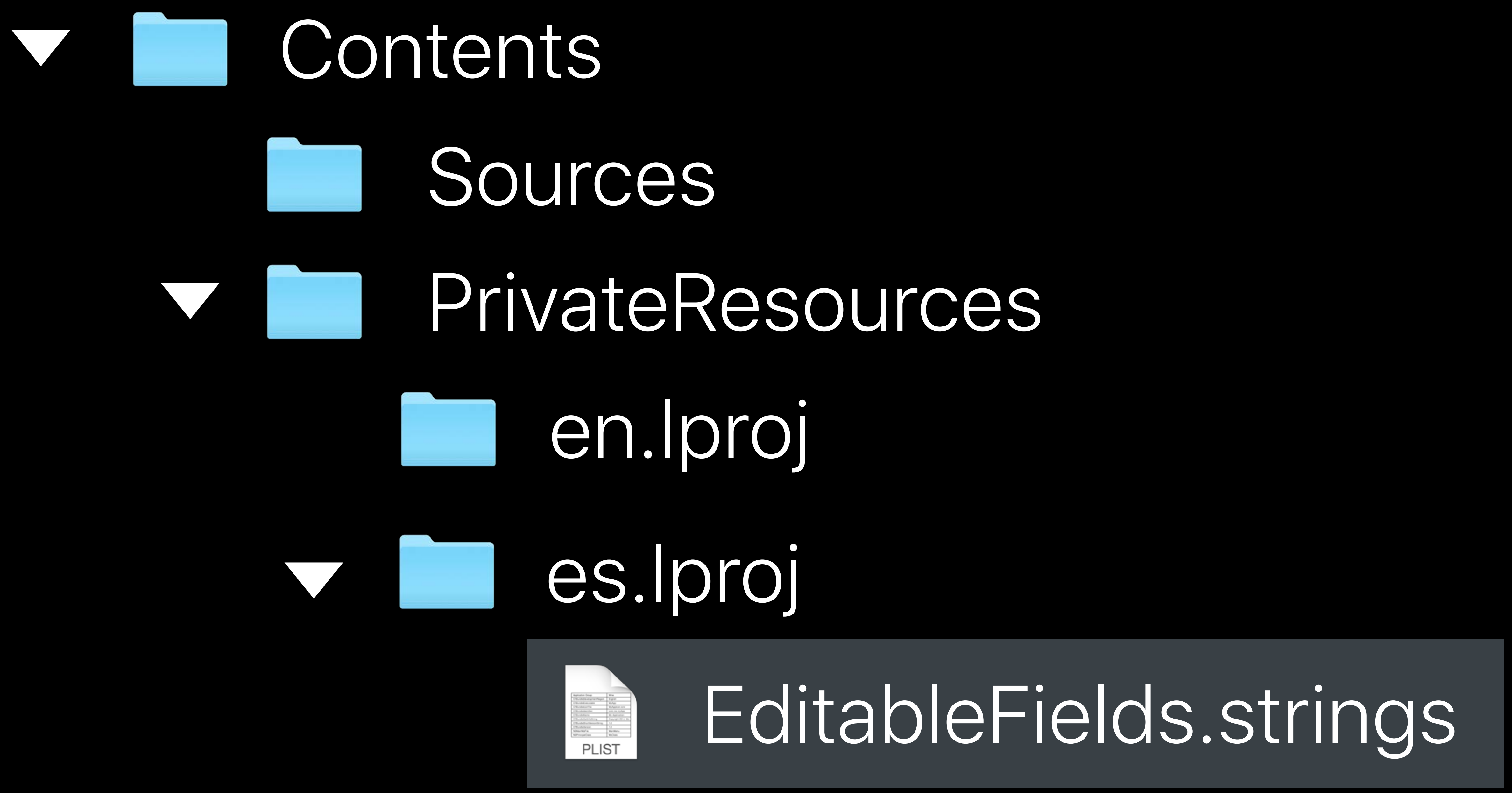
*.strings files

Contents.swift

```
//#-editable-code Tap to enter code
```

es.lproj/EditableFields.strings

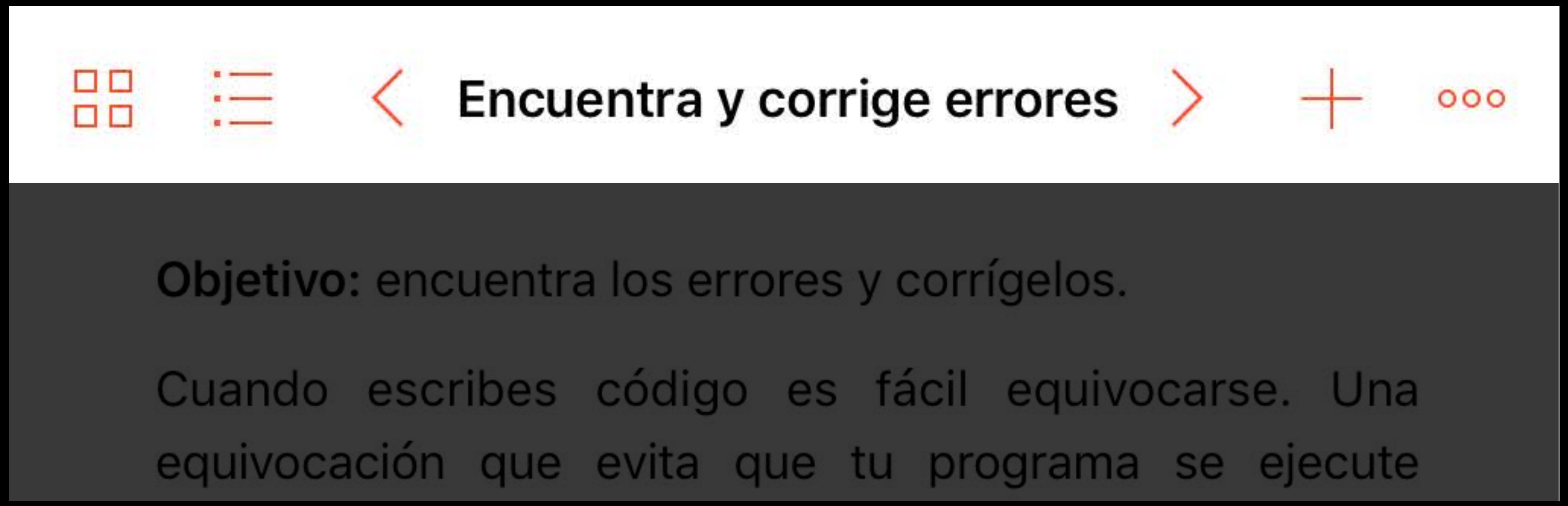
```
"Tap to enter code" = "Toca para ingresar código";
```



- Contents
  - Sources
  - PrivateResources
    - en.lproj
    - es.lproj
      - EditableFields.strings

Page and Chapter Titles

*.strings files



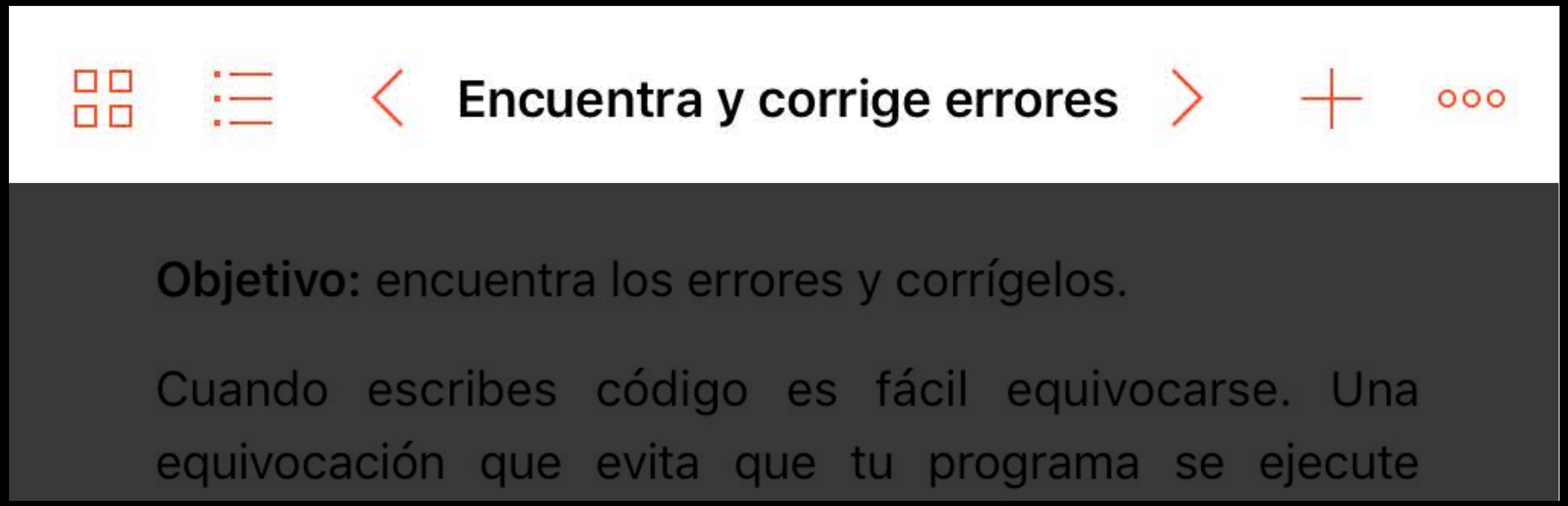
Encuentra y corrige errores

Objetivo: encuentra los errores y corrígelos.

Cuando escribes código es fácil equivocarse. Una equivocación que evita que tu programa se ejecute

Page and Chapter Titles

*.strings files



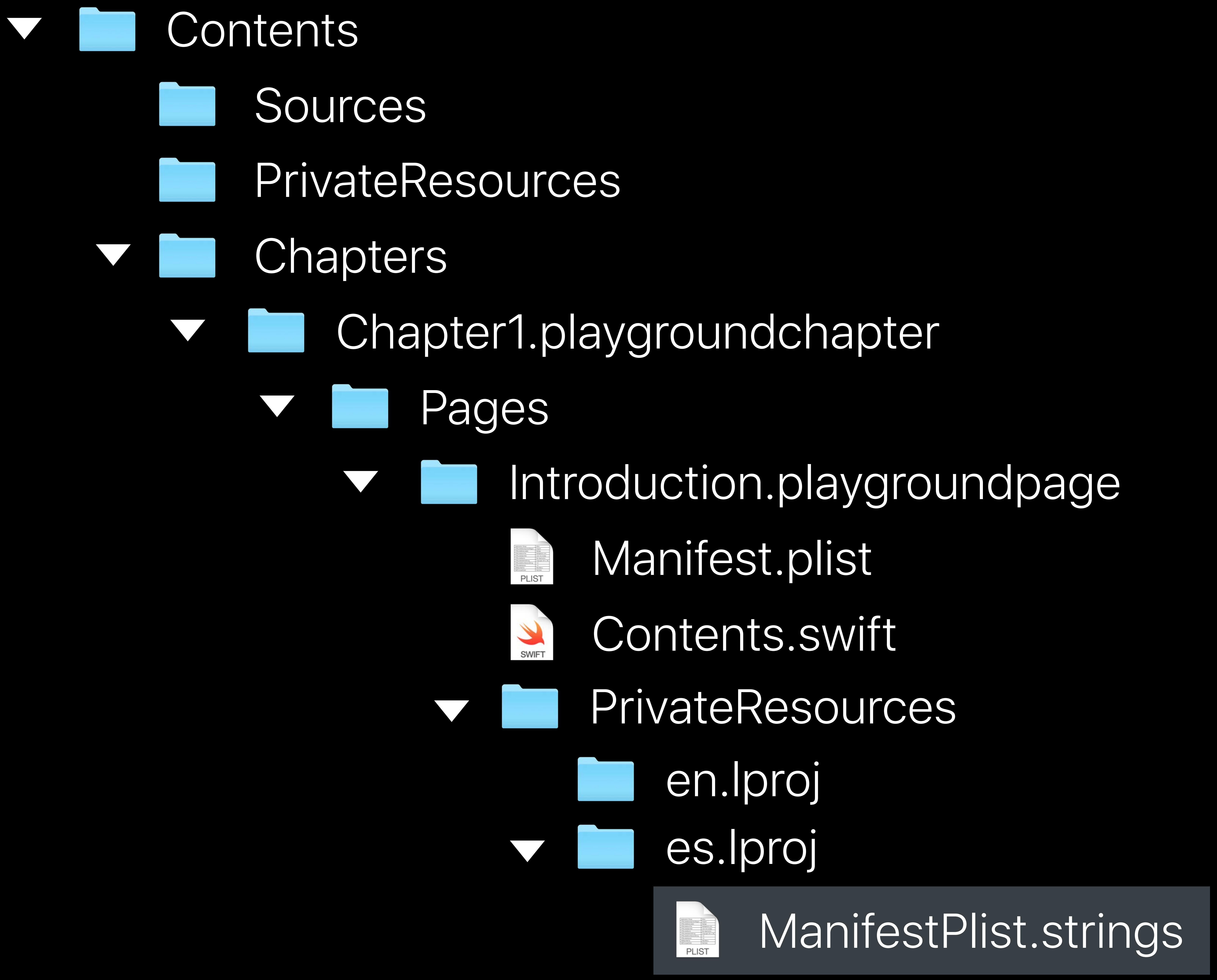
Encuentra y corrige errores

Objetivo: encuentra los errores y corrígelos.

Cuando escribes código es fácil equivocarse. Una equivocación que evita que tu programa se ejecute

es.lproj/ManifestPlist.strings

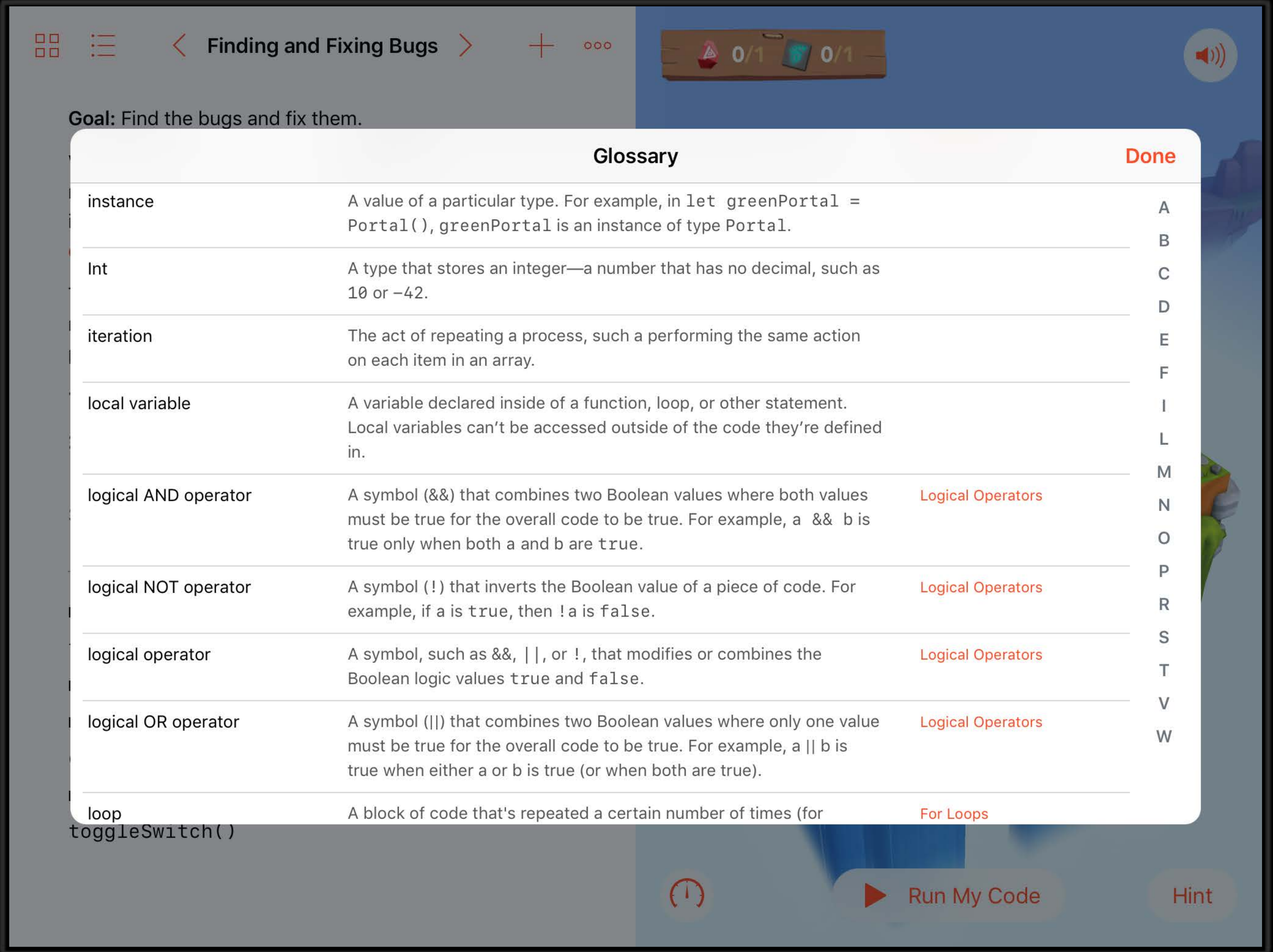
```
Name = "Encuentra y corrige errores";
```



```
Contents
    Sources
    PrivateResources
    Chapters
        Chapter1.playgroundchapter
            Pages
                Introduction.playgroundpage
                    Manifest.plist
                    Contents.swift
                    PrivateResources
                        en.lproj
                        es.lproj
                            ManifestPlist.strings
```

Glossary

*.plist files



Goal: Find the bugs and fix them.

instance — A value of a particular type. For example, in `let greenPortal = Portal()`, `greenPortal` is an instance of type `Portal`.

Int — A type that stores an integer—a number that has no decimal, such as `10` or `−42`.

iteration — The act of repeating a process, such a performing the same action on each item in an array.

local variable — A variable declared inside of a function, loop, or other statement. Local variables can't be accessed outside of the code they're defined in.

logical AND operator — A symbol (&&) that combines two Boolean values where both values must be true for the overall code to be true. For example, a `&&` b is true only when both a and b are `true`. — Logical Operators

logical NOT operator — A symbol (`!`) that inverts the Boolean value of a piece of code. For example, if a is `true`, then `!`a is `false`. — Logical Operators

logical operator — A symbol, such as &&, `||`, or `!`, that modifies or combines the Boolean logic values `true` and `false`. — Logical Operators

logical OR operator — A symbol (||) that combines two Boolean values where only one value must be true for the overall code to be true. For example, a || b is true when either a or b is true (or when both are true). — Logical Operators

loop — A block of code that's repeated a certain number of times (for — For Loops

```
toggleSwitch()
```

en.lproj/Glossary.plist

```
...
<key>command</key>
        <dict>
            <key>FirstUse</key>
            <dict>
                <key>PageReference</key>
                <string>Document1.playgroundchapter/Exercise1.playgroundpage</string>
                <key>Title</key>
                <string>Commands</string>
            </dict>
            <key>Definition</key>
            <string>Code that tells an application to perform a specific action.
            </string>
            <key>Title</key>
            <string>command</string>
        </dict>
```

en.lproj/Glossary.plist

```
...
<key>command</key>
        <dict>
            <key>FirstUse</key>
            <dict>
                <key>PageReference</key>
                <string>Document1.playgroundchapter/Exercise1.playgroundpage</string>
                <key>Title</key>
                <string>Commands</string>
            </dict>
            <key>Definition</key>
            <string>Code that tells an application to perform a specific action.
            </string>
            <key>Title</key>
            <string>command</string>
        </dict>
```

en.lproj/Glossary.plist

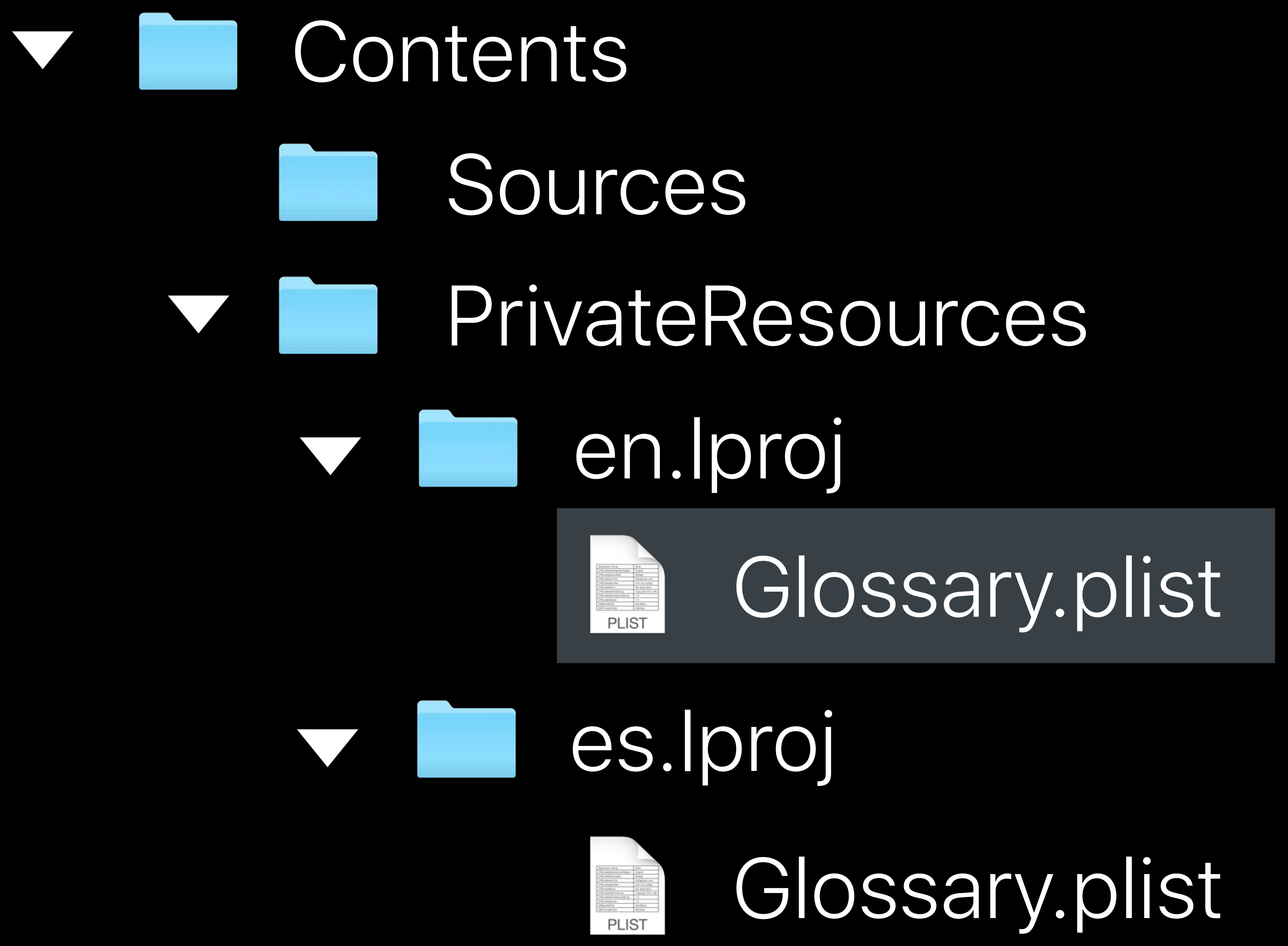
```
...
<key>command</key>
        <dict>
            <key>FirstUse</key>
            <dict>
                <key>PageReference</key>
                <string>Document1.playgroundchapter/Exercise1.playgroundpage</string>
                <key>Title</key>
                <string>Commands</string>
            </dict>
            <key>Definition</key>
            <string>Code that tells an application to perform a specific action.
            </string>
            <key>Title</key>
            <string>command</string>
        </dict>
```

en.lproj/Glossary.plist

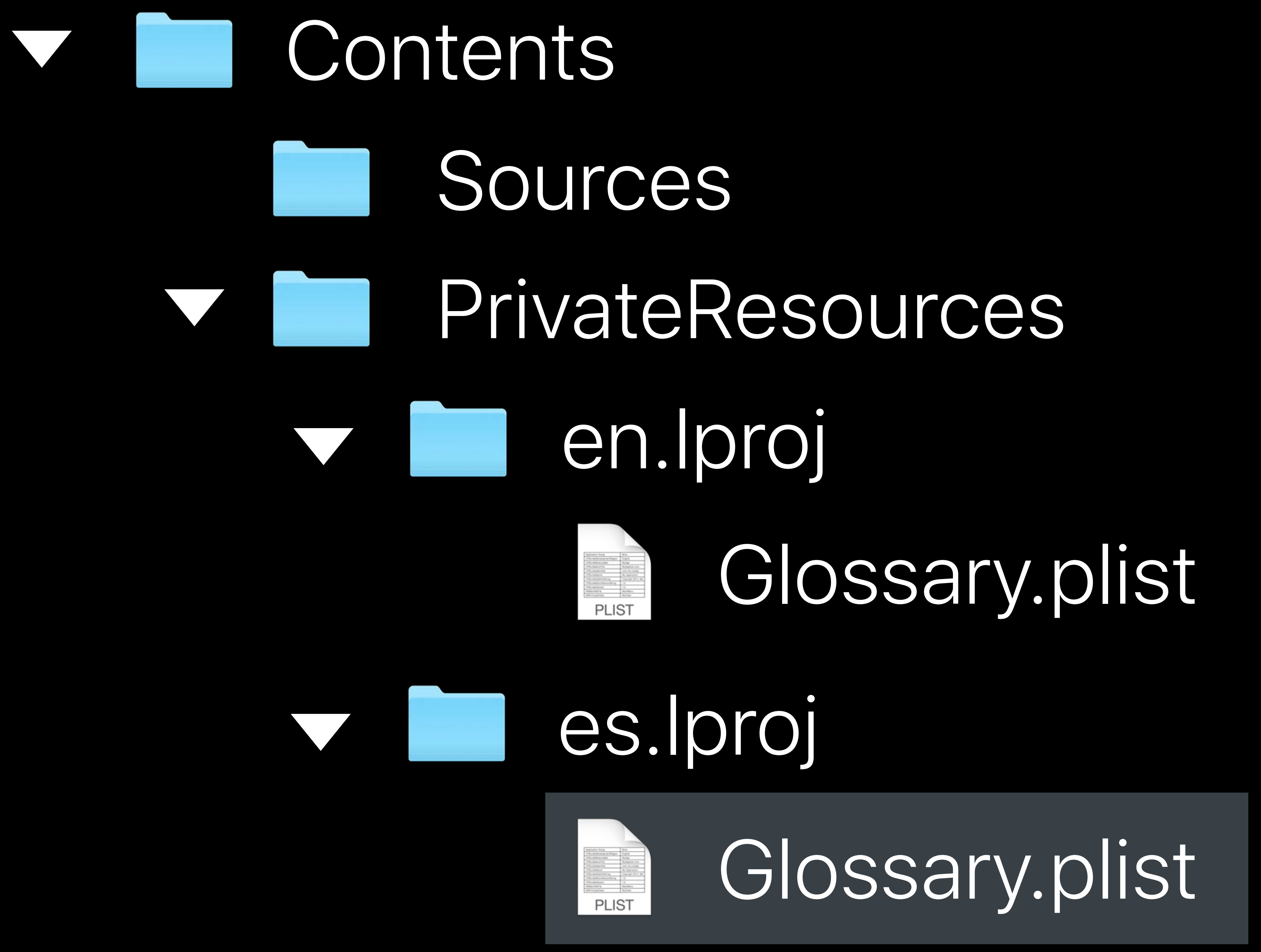
```
...
<key>command</key>
        <dict>
            <key>FirstUse</key>
            <dict>
                <key>PageReference</key>
                <string>Document1.playgroundchapter/Exercise1.playgroundpage</string>
                <key>Title</key>
                <string>Commands</string>
            </dict>
            <key>Definition</key>
            <string>Code that tells an application to perform a specific action.
            </string>
            <key>Title</key>
            <string>command</string>
        </dict>
```

es.lproj/Glossary.plist

```
...
<key>command</key>
        <dict>
            <key>FirstUse</key>
            <dict>
                <key>PageReference</key>
                <string>Document1.playgroundchapter/Exercise1.playgroundpage</string>
                <key>Title</key>
                <string>Comandos</string>
            </dict>
            <key>Definition</key>
            <string>Código que le indica a una aplicación que realice una acción específica.
            </string>
            <key>Title</key>
            <string>comando</string>
        </dict>
```



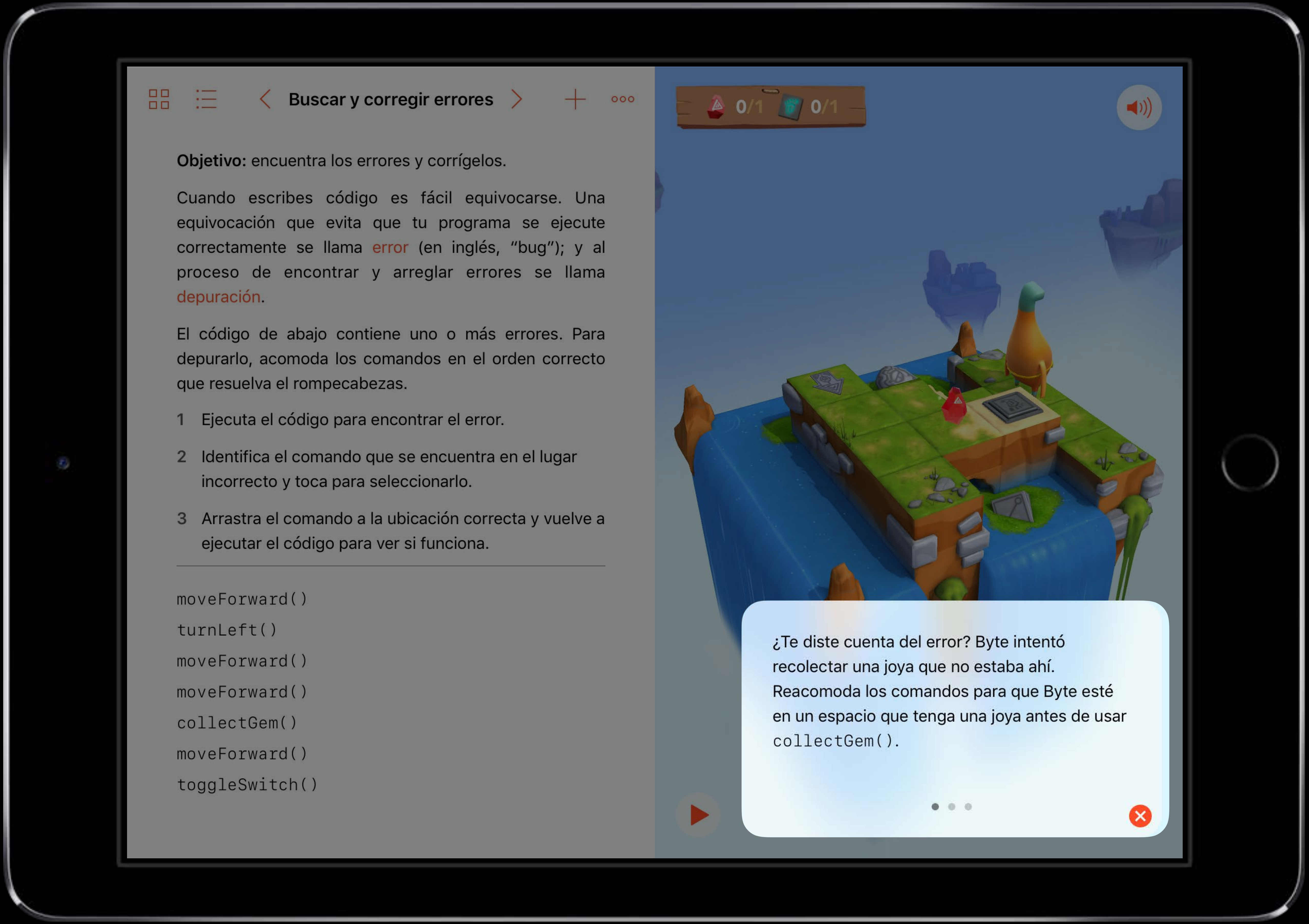
- Contents
  - Sources
  - PrivateResources
    - en.lproj
      - Glossary.plist
    - es.lproj
      - Glossary.plist



```
Contents
    Sources
    PrivateResources
        en.lproj
            Glossary.plist
        es.lproj
            Glossary.plist
```

Hints

*.plist files



Objetivo: encuentra los errores y corrígelos.

Cuando escribes código es fácil equivocarse. Una equivocación que evita que tu programa se ejecute correctamente se llama error (en inglés, "bug"); y al proceso de encontrar y arreglar errores se llama depuración.

El código de abajo contiene uno o más errores. Para depurarlo, acomoda los comandos en el orden correcto que resuelva el rompecabezas.

1 Ejecuta el código para encontrar el error.

2 Identifica el comando que se encuentra en el lugar incorrecto y toca para seleccionarlo.

3 Arrastra el comando a la ubicación correcta y vuelve a ejecutar el código para ver si funciona.

```
moveForward()
turnLeft()
moveForward()
moveForward()
collectGem()
moveForward()
toggleSwitch()
```

¿Te diste cuenta del error? Byte intentó recolectar una joya que no estaba ahí. Reacomoda los comandos para que Byte esté en un espacio que tenga una joya antes de usar `collectGem()`.

en.lproj/Hints.plist

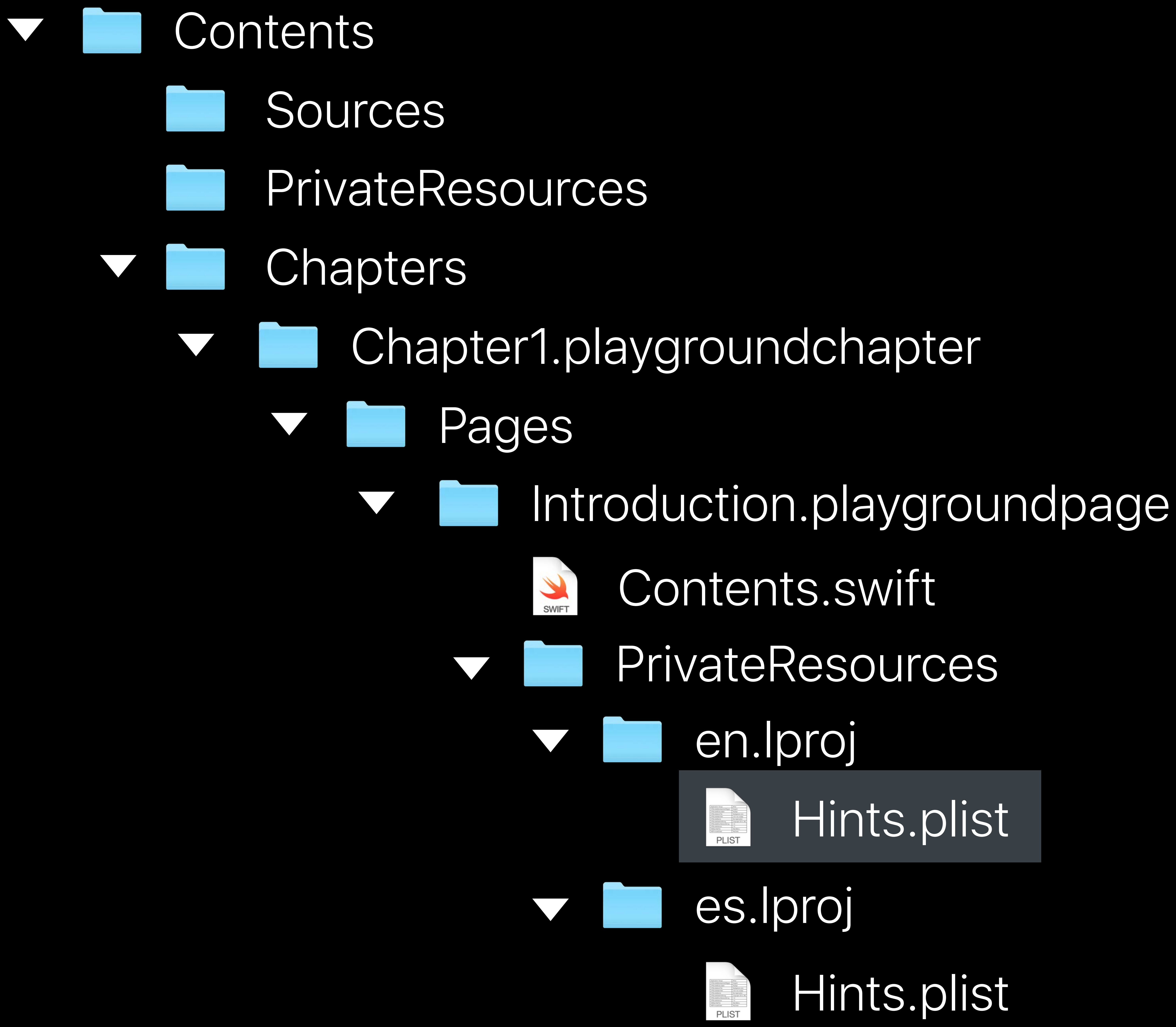
```
<dict>
    <key>Hints</key>
    <array>
        <dict>
            <key>Content</key>
            <string>This is an inline hint.</string>
        </dict>
        <dict>
            <key>Content</key>
            <string>This is an inline hint with a spoiler button.</string>
            <key>SpoilerButtonTitle</key>
            <string>Show Spoiler</string>
        </dict>
    </array>
</dict>
```

en.lproj/Hints.plist

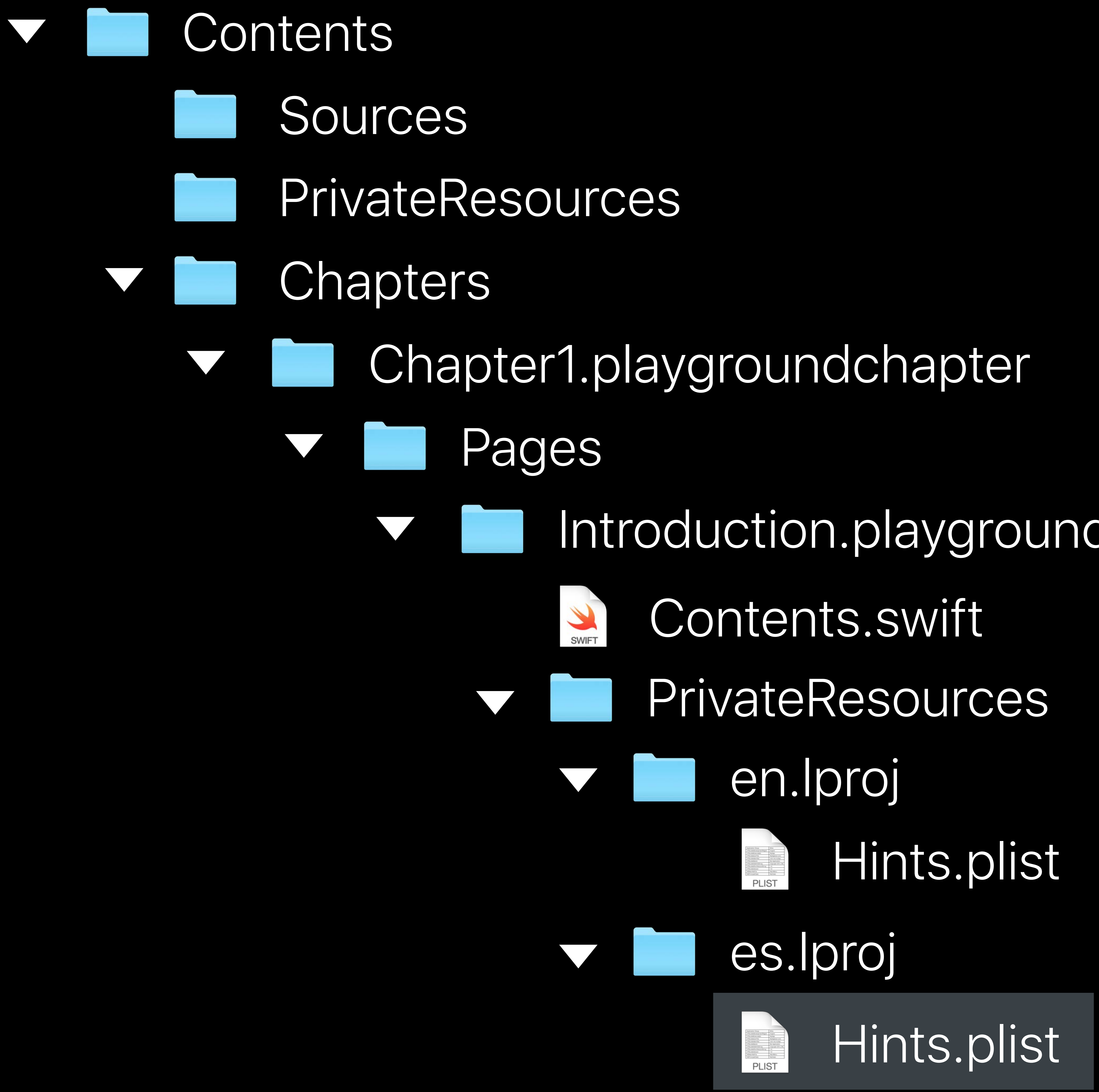
```
<dict>
    <key>Hints</key>
    <array>
        <dict>
            <key>Content</key>
            <string>This is an inline hint.</string>
        </dict>
        <dict>
            <key>Content</key>
            <string>This is an inline hint with a spoiler button.</string>
            <key>SpoilerButtonTitle</key>
            <string>Show Spoiler</string>
        </dict>
    </array>
</dict>
```

es.lproj/Hints.plist

```
<dict>
    <key>Hints</key>
    <array>
        <dict>
            <key>Content</key>
            <string>Esta es una sugerencia en línea.</string>
        </dict>
        <dict>
            <key>Content</key>
            <string>Esta es una sugerencia en línea con un botón spoiler.</string>
            <key>SpoilerButtonTitle</key>
            <string>Mostrar Spoiler</string>
        </dict>
    </array>
</dict>
```



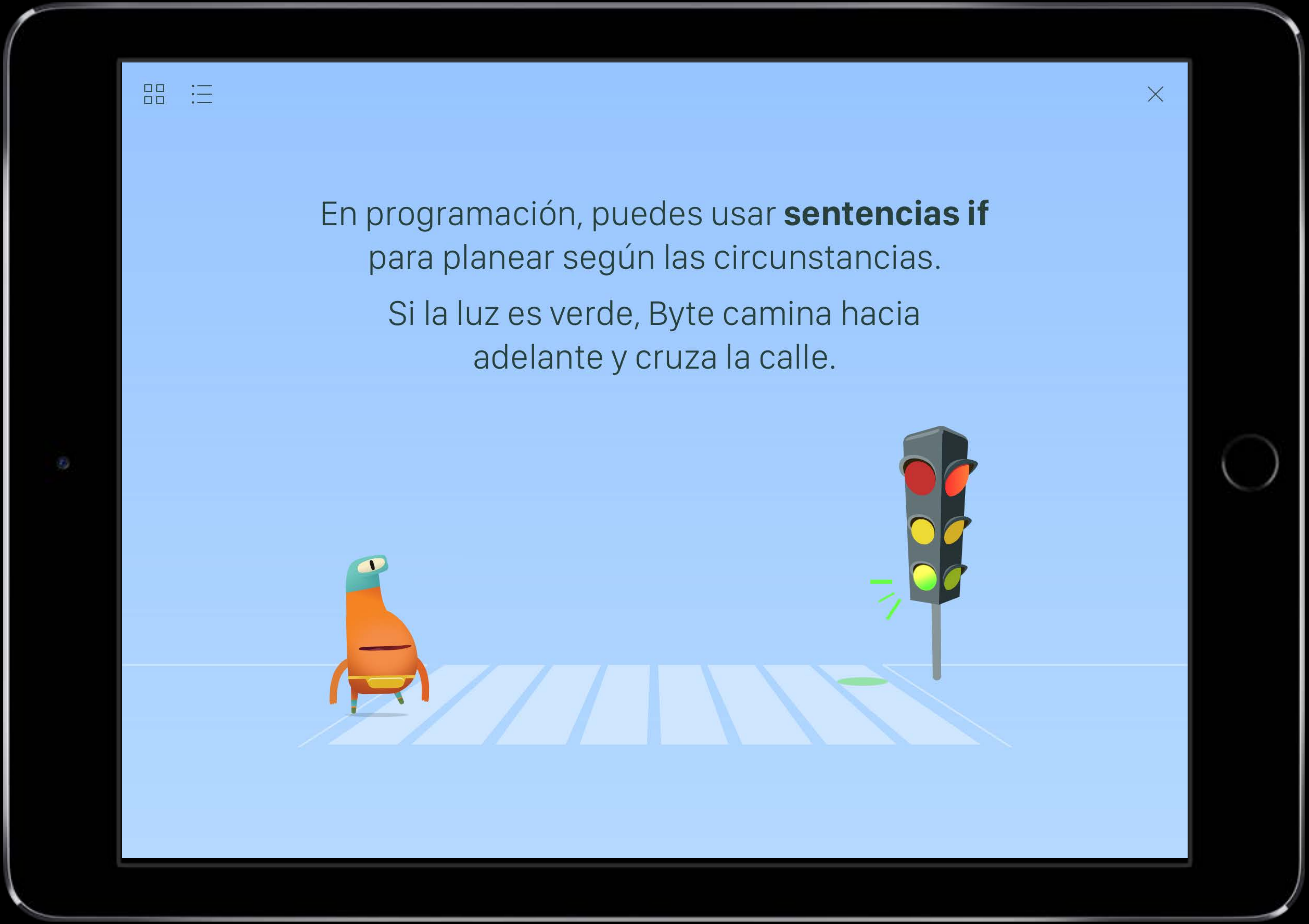
```
Contents
    Sources
    PrivateResources
    Chapters
        Chapter1.playgroundchapter
            Pages
                Introduction.playgroundpage
                    Contents.swift
                    PrivateResources
                        en.lproj
                            Hints.plist
                        es.lproj
                            Hints.plist
```



```
Contents
    Sources
    PrivateResources
    Chapters
        Chapter1.playgroundchapter
            Pages
                Introduction.playgroundpage
                    Contents.swift
                    PrivateResources
                        en.lproj
                            Hints.plist
                        es.lproj
                            Hints.plist
```

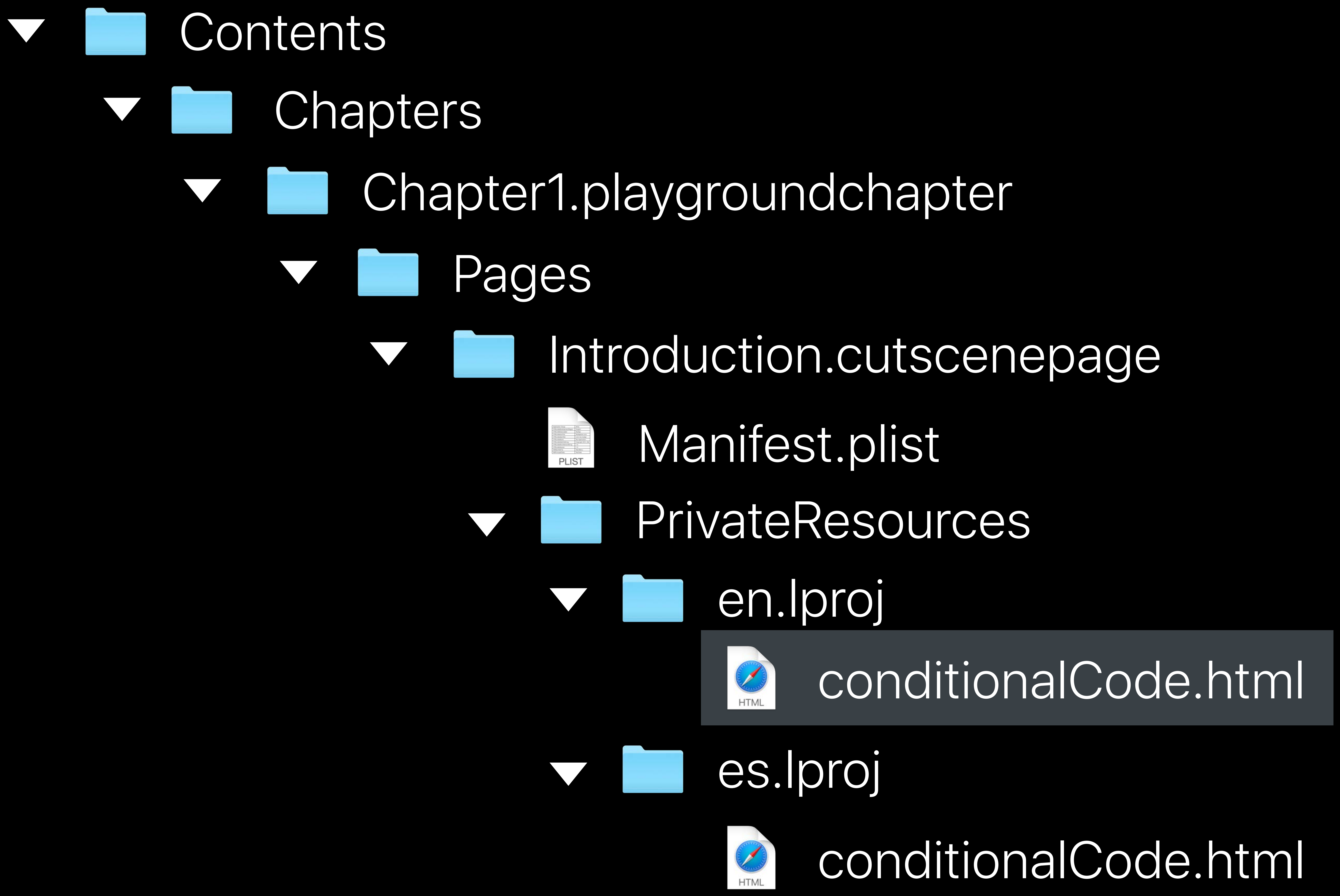
Cutscenes

*.html files

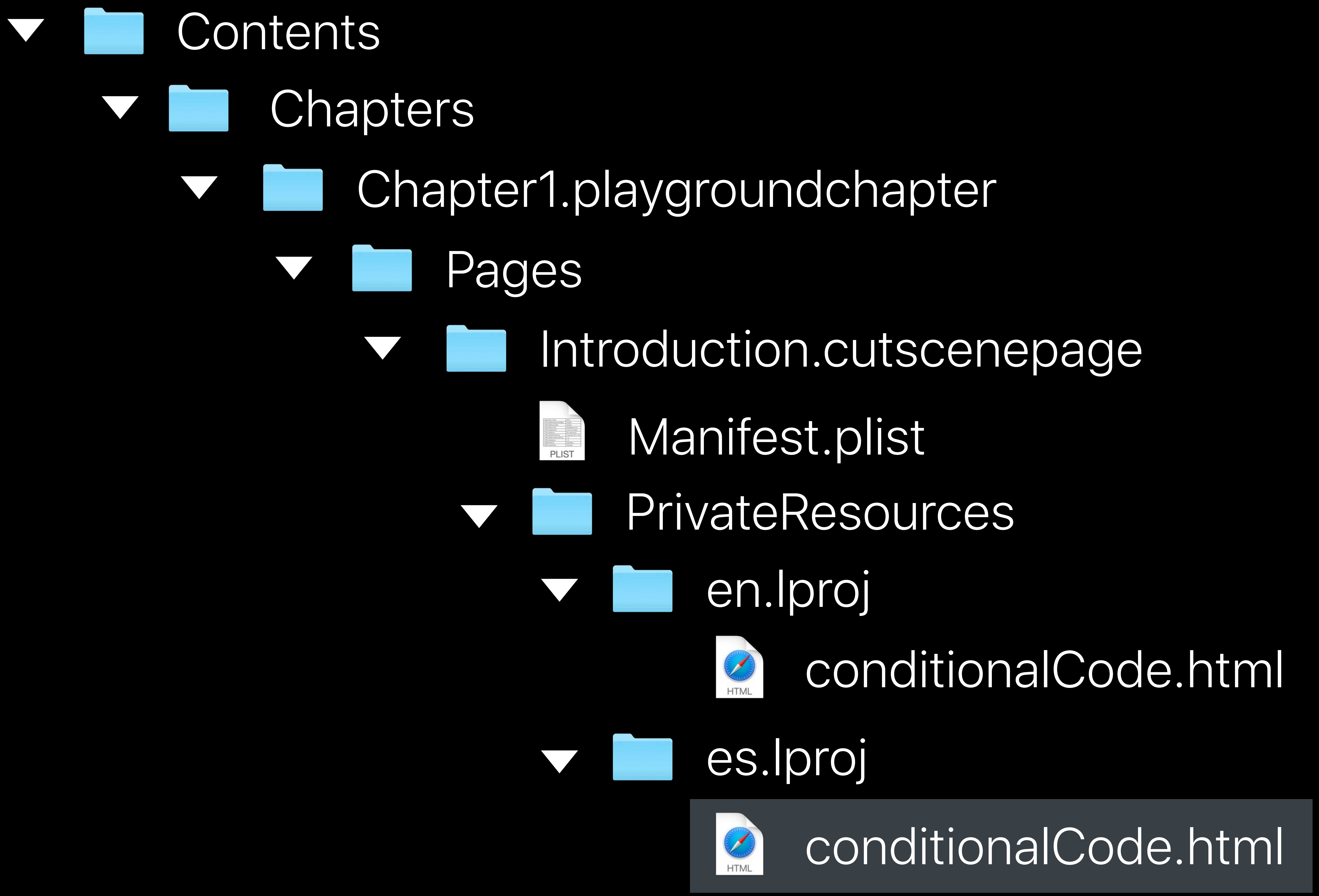


En programación, puedes usar **sentencias if** para planear según las circunstancias.

Si la luz es verde, Byte camina hacia adelante y cruza la calle.



```
Contents
  Chapters
    Chapter1.playgroundchapter
      Pages
        Introduction.cutscenepage
          Manifest.plist
          PrivateResources
            en.lproj
              conditionalCode.html
            es.lproj
              conditionalCode.html
```



```
Contents
  Chapters
    Chapter1.playgroundchapter
      Pages
        Introduction.cutscenepage
          Manifest.plist
          PrivateResources
            en.lproj
              conditionalCode.html
            es.lproj
              conditionalCode.html
```

Summary

Summary

Understand your audience

Summary

Understand your audience

Nuanced and complex

Summary

Understand your audience

Nuanced and complex

Localization is a part of design

More Information

https://developer.apple.com/wwdc17/410

Related Sessions

Localizing with Xcode 9		WWDC 2017
SceneKit in Swift Playgrounds		WWDC 2017
What's New in Swift Playgrounds		WWDC 2017
Teaching with Swift Playgrounds	Hall 2	Friday 2:50PM

Labs

Internationalization Lab	Technology Lab I	Fri 9:00AM–11:00AM
Swift Open Hours	Technology Lab D	Fri 12:00PM–1:30PM
Xcode Open Hours	Technology Lab K	Fri 1:50PM–4:00PM

WWDC17