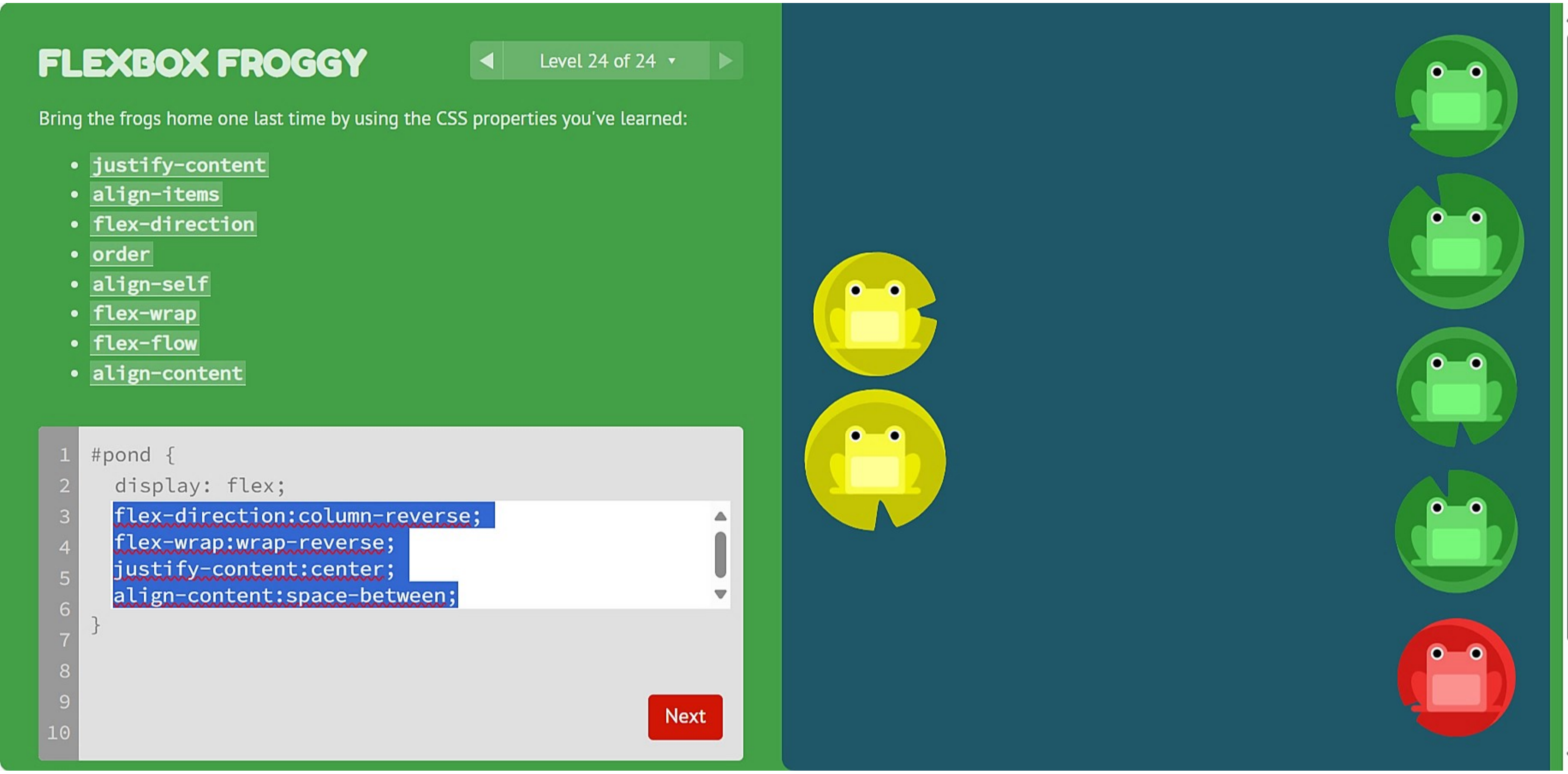
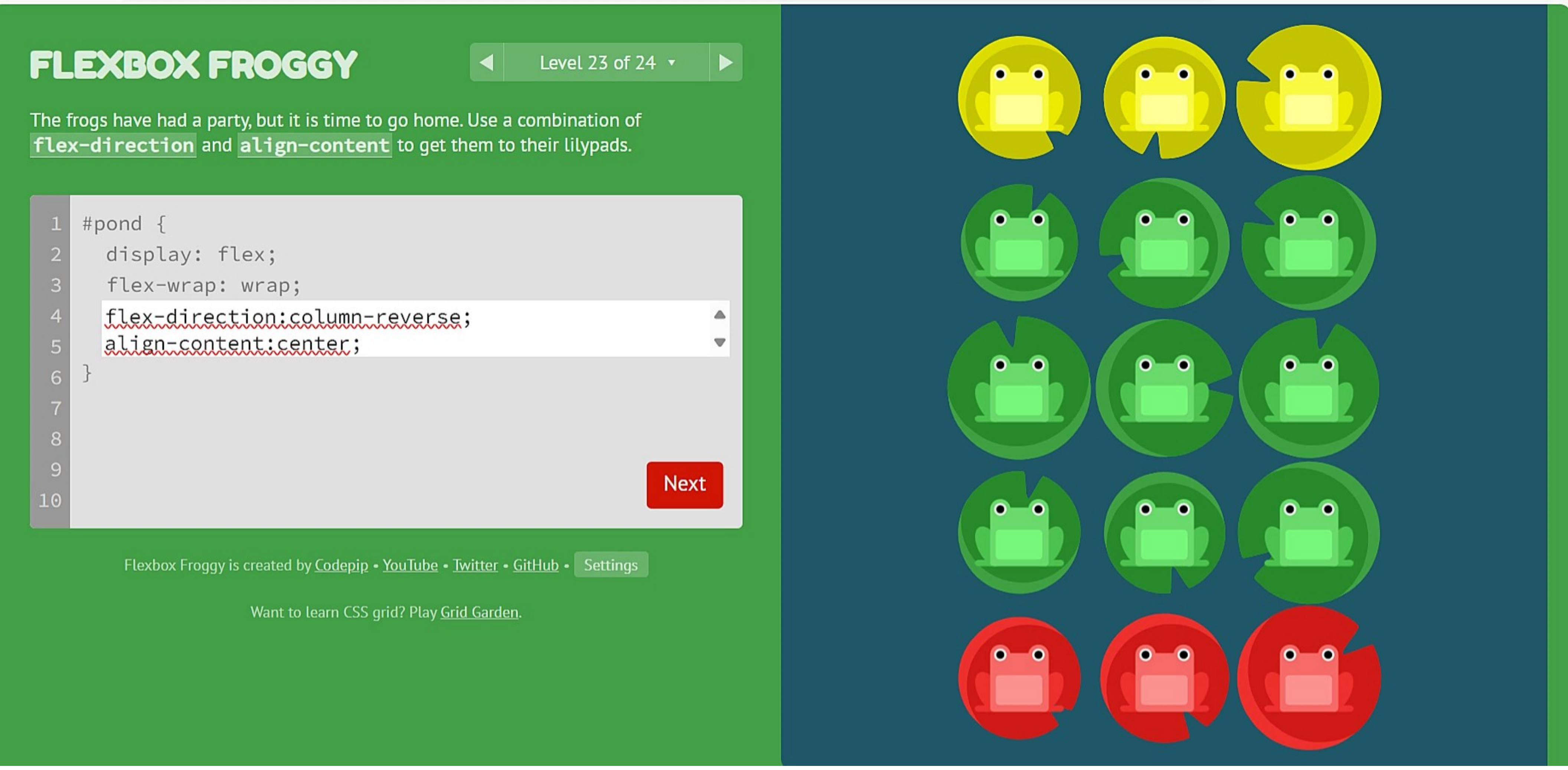


Bring the frogs home one last time by using the CSS properties you've learned:

- `justify-content`
- `align-items`
- `flex-direction`
- `order`
- `align-self`
- `flex-wrap`
- `flex-flow`
- `align-content`

```css
#pond {
  display: flex;
  flex-direction:column-reverse;
  flex-wrap:wrap-reverse;
  justify-content:center;
  align-content:space-between;
}
```

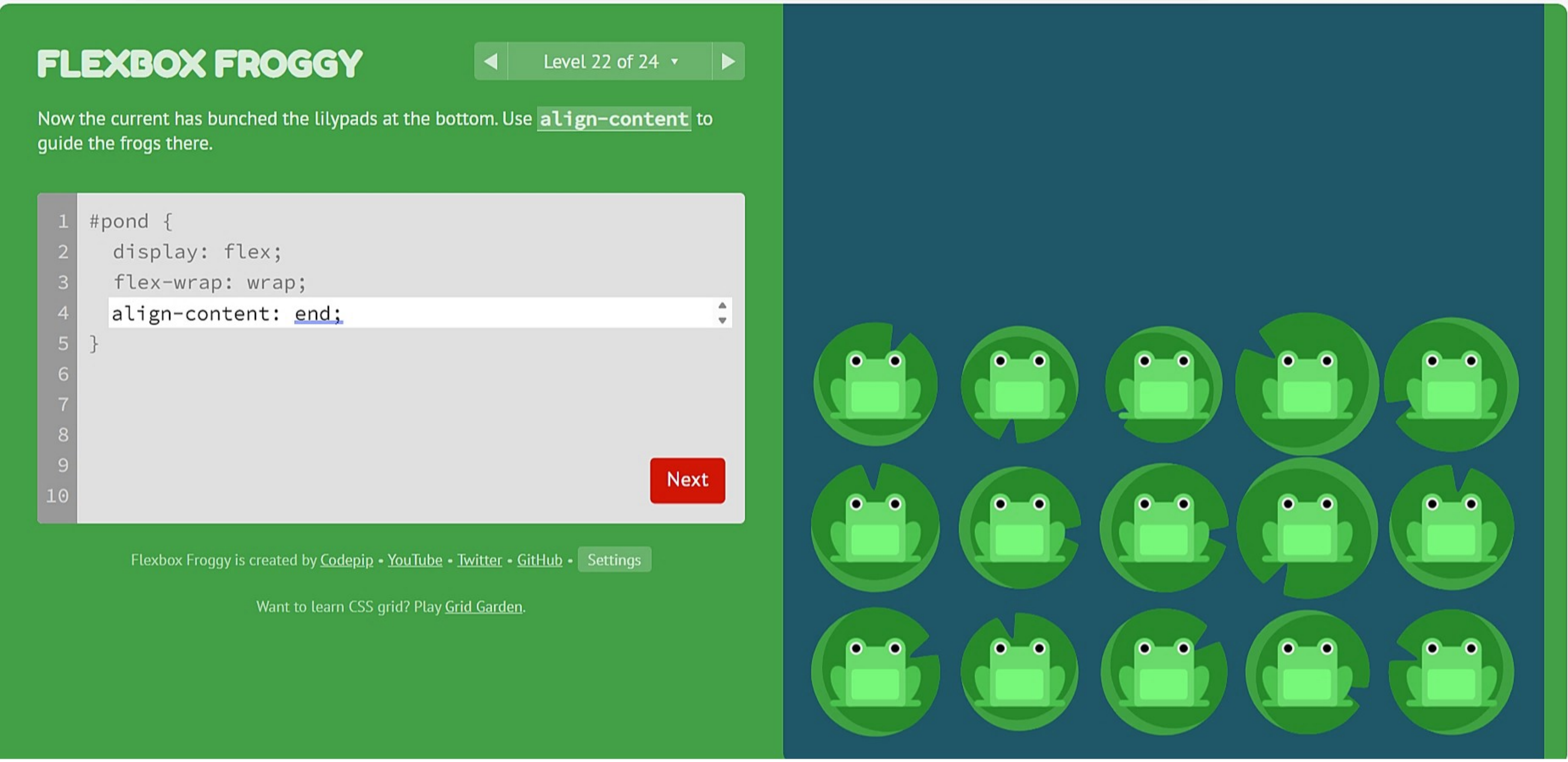


The frogs have had a party, but it is time to go home. Use a combination of `flex-direction` and `align-content` to get them to their lilypads.

```
#pond {
  display: flex;
  flex-wrap: wrap;
  flex-direction:column-reverse;
  align-content:center;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.



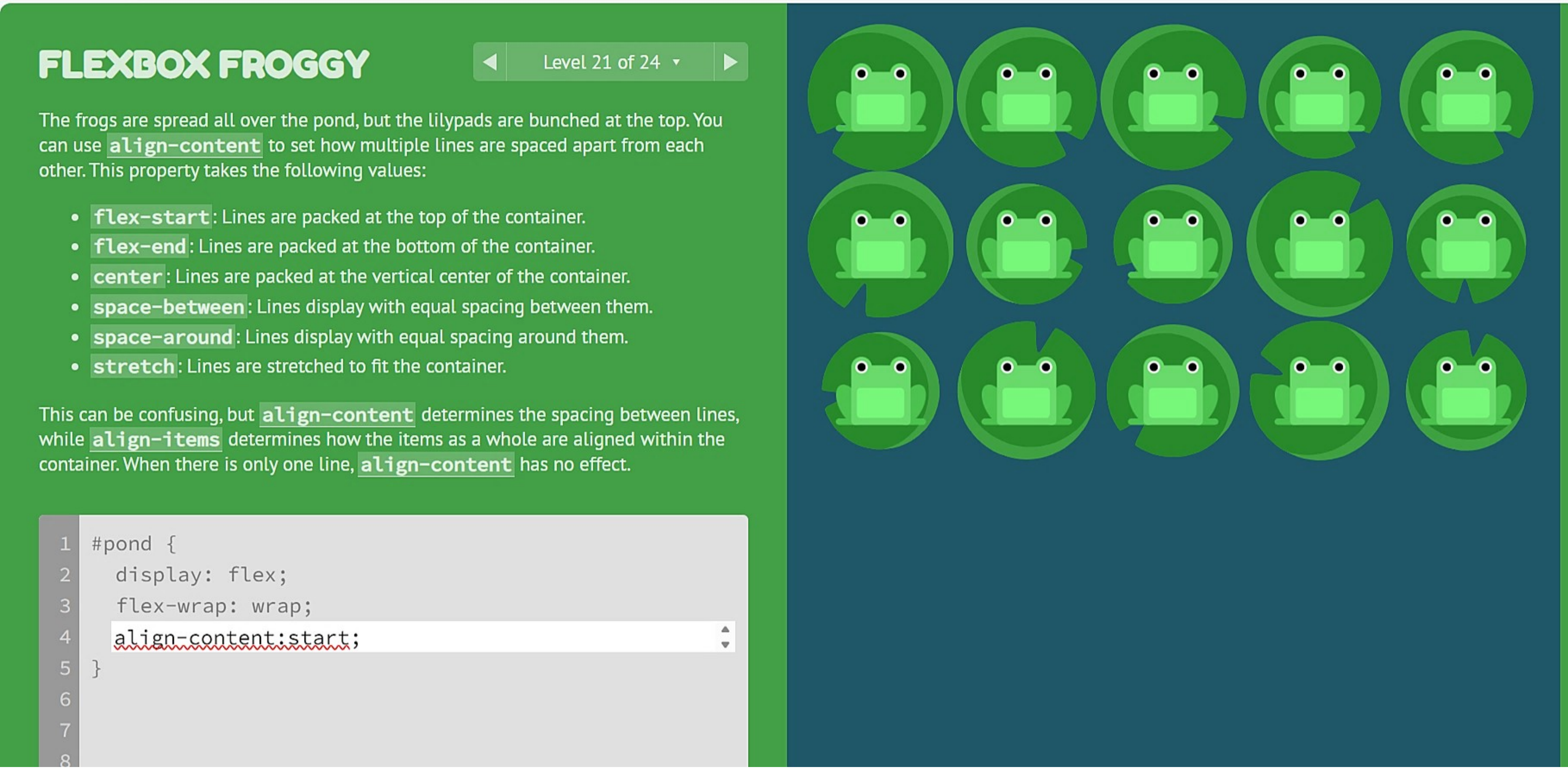
FLEXBOX FROGGY

Now the current has bunched the lilypads at the bottom. Use `align-content` to guide the frogs there.

```
#pond {
  display: flex;
  flex-wrap: wrap;
  align-content: end;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY

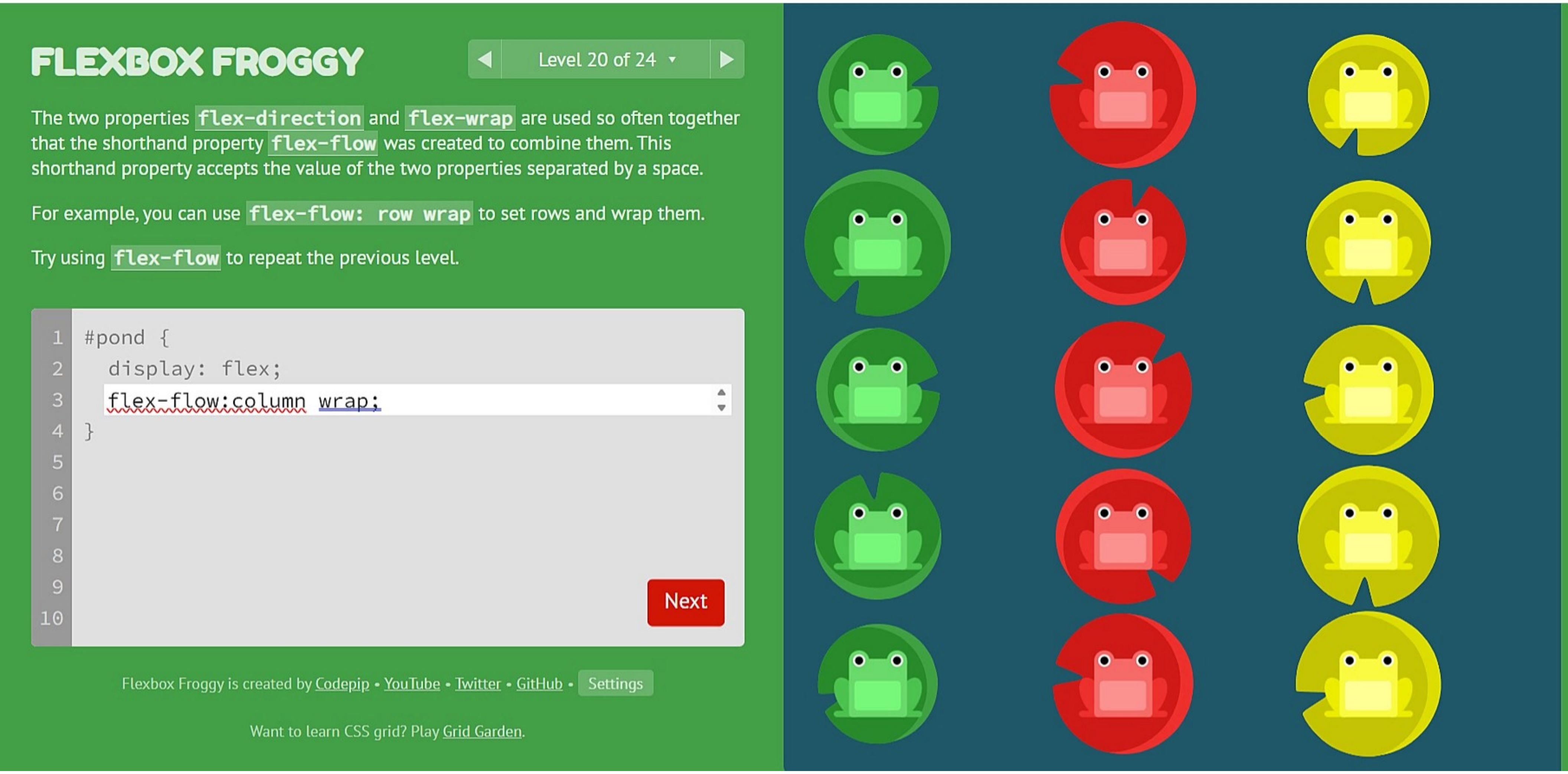
Level 21 of 24

The frogs are spread all over the pond, but the lilypads are bunched at the top. You can use `align-content` to set how multiple lines are spaced apart from each other. This property takes the following values:

- `flex-start`: Lines are packed at the top of the container.
- `flex-end`: Lines are packed at the bottom of the container.
- `center`: Lines are packed at the vertical center of the container.
- `space-between`: Lines display with equal spacing between them.
- `space-around`: Lines display with equal spacing around them.
- `stretch`: Lines are stretched to fit the container.

This can be confusing, but `align-content` determines the spacing between lines, while `align-items` determines how the items as a whole are aligned within the container. When there is only one line, `align-content` has no effect.

```
#pond {
  display: flex;
  flex-wrap: wrap;
  align-content:start;
}
```



FLEXBOX FROGGY

The two properties `flex-direction` and `flex-wrap` are used so often together that the shorthand property `flex-flow` was created to combine them. This shorthand property accepts the value of the two properties separated by a space.

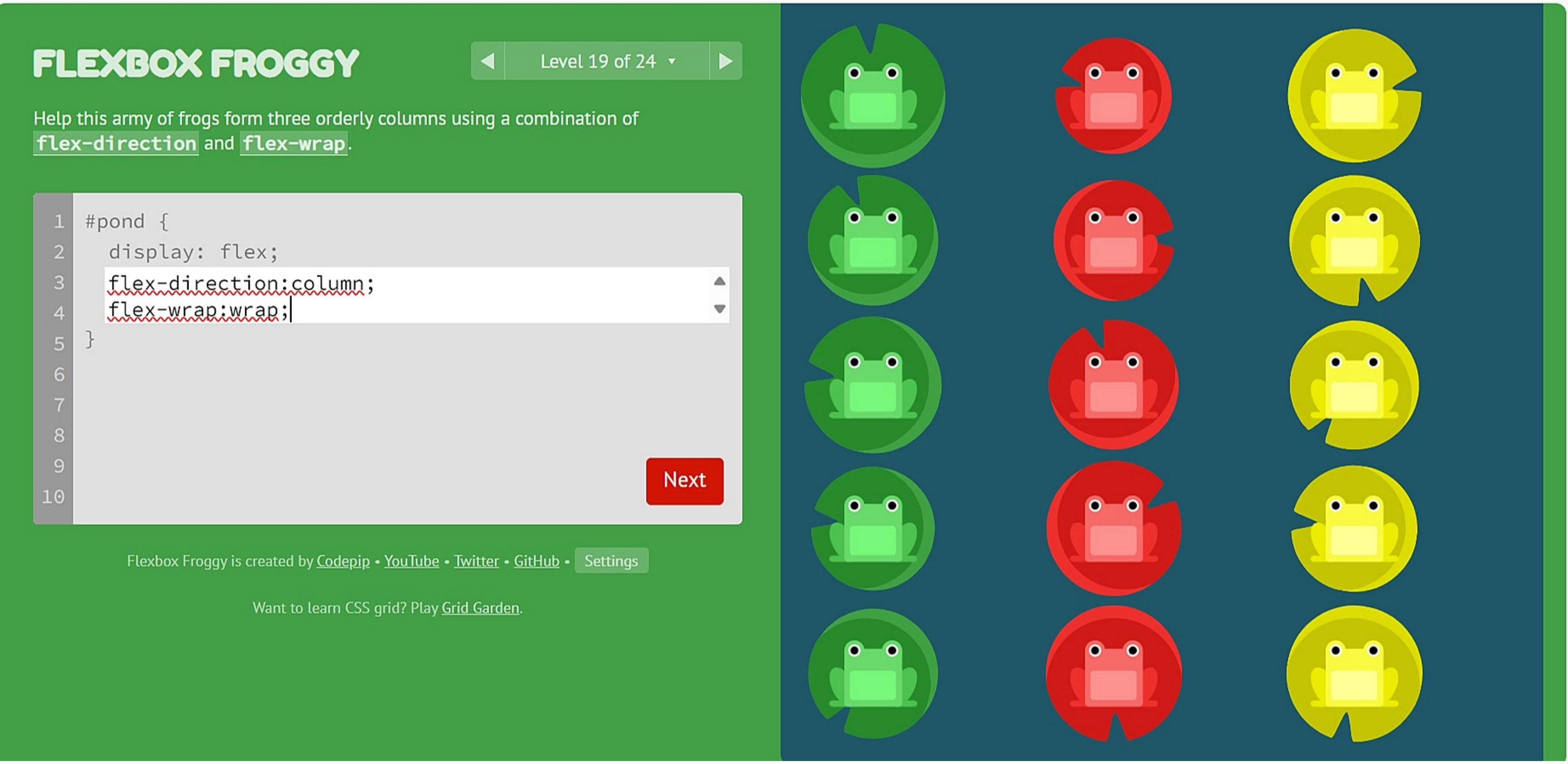
For example, you can use `flex-flow: row wrap` to set rows and wrap them.

Try using `flex-flow` to repeat the previous level.

```
#pond {
  display: flex;
  flex-flow:column wrap;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.

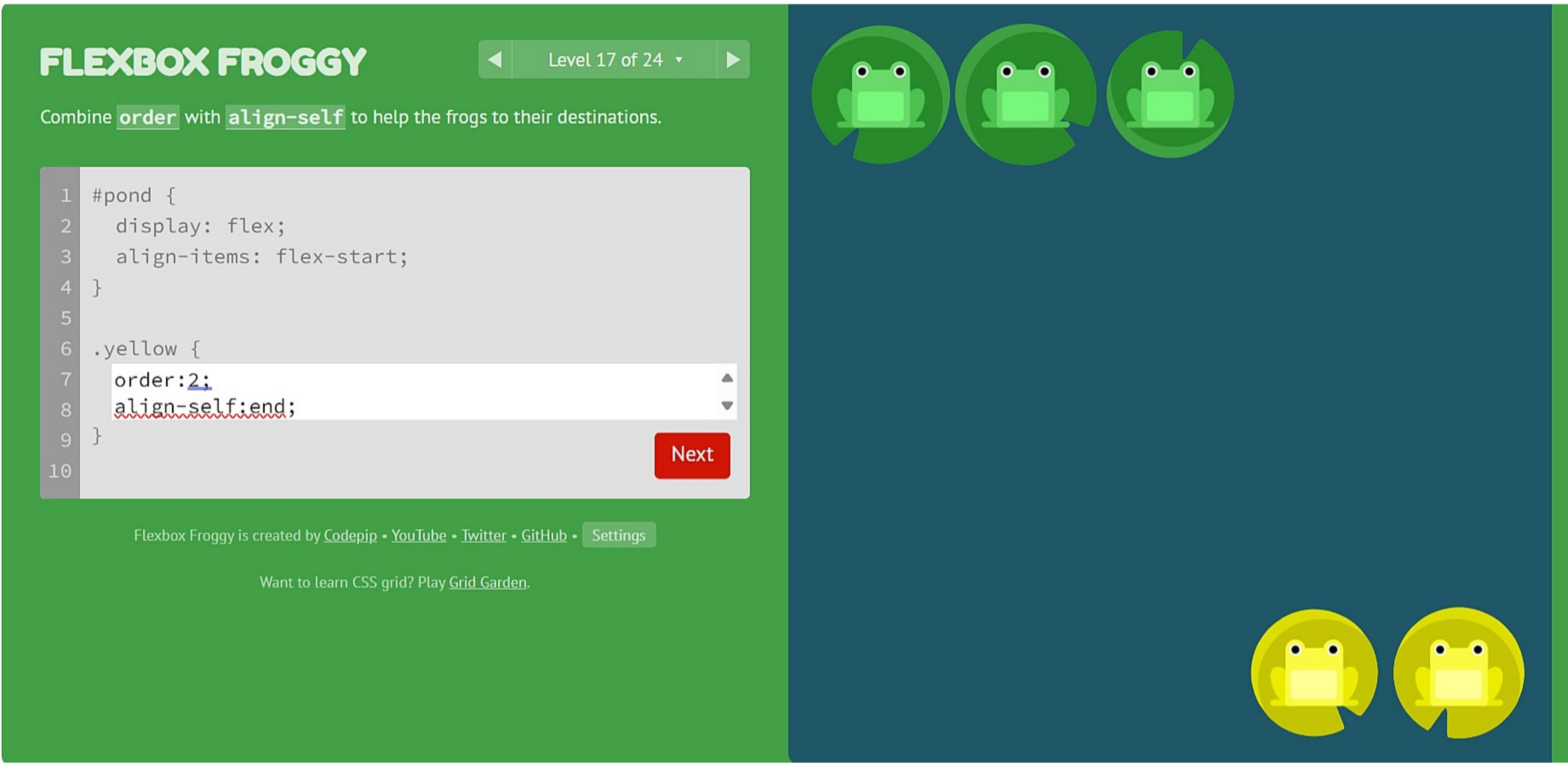


Help this army of frogs form three orderly columns using a combination of `flex-direction` and `flex-wrap`.

```
#pond {
  display: flex;
  flex-direction:column;
  flex-wrap:wrap;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub •

Want to learn CSS grid? Play Grid Garden.



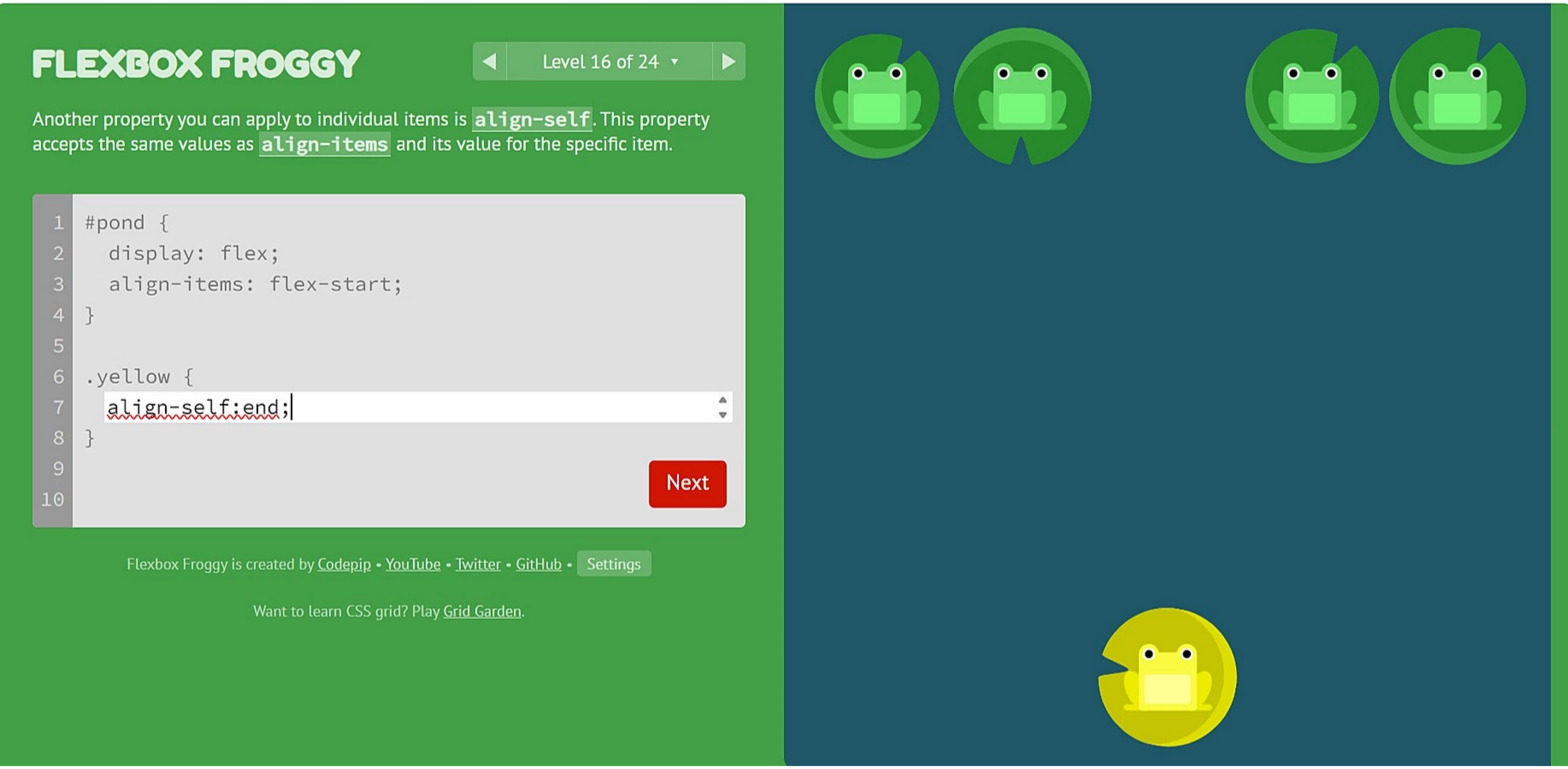
Combine `order` with `align-self` to help the frogs to their destinations.

```
#pond {
  display: flex;
  align-items: flex-start;
}

.yellow {
  order:2;
  align-self:end;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub •

Want to learn CSS grid? Play Grid Garden.

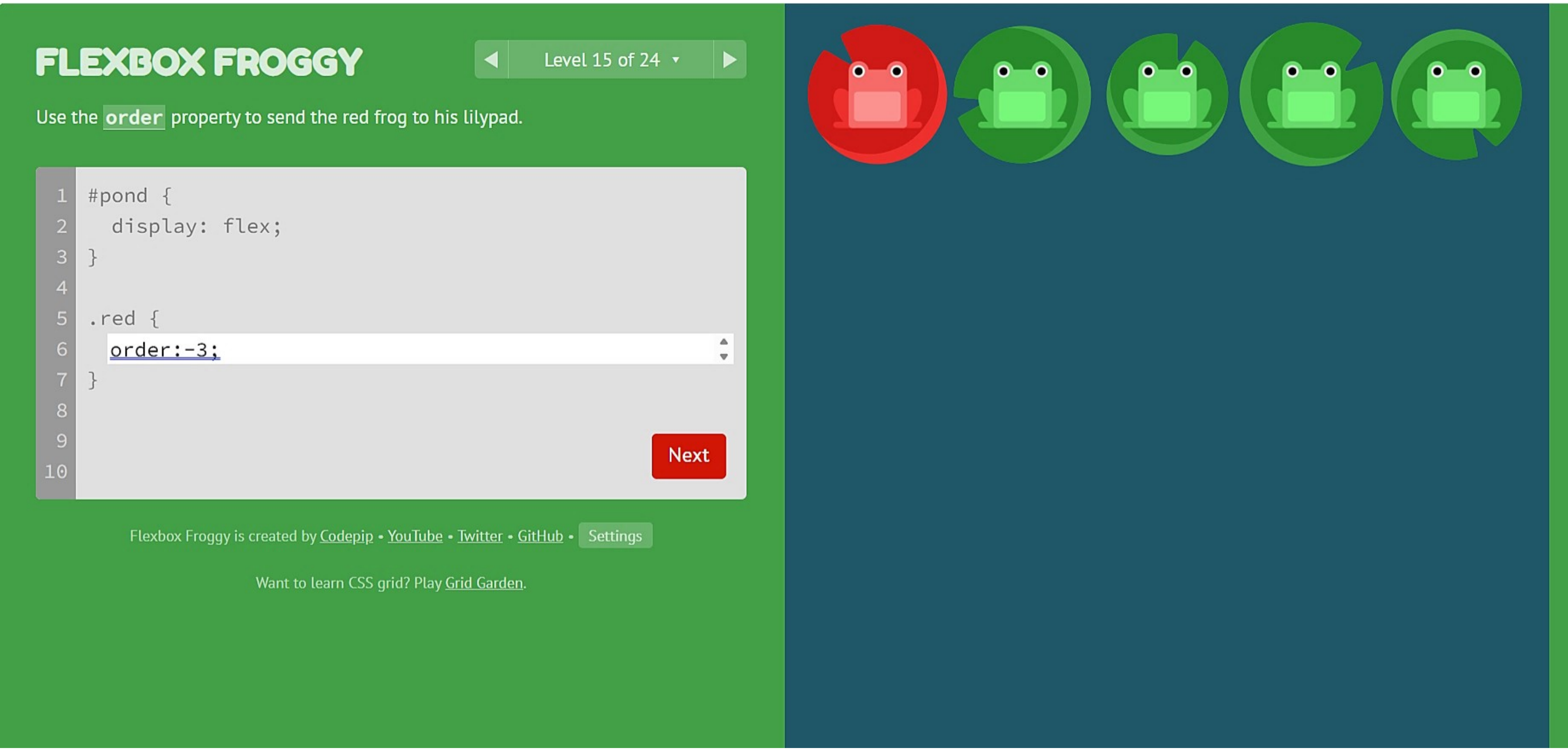

FLEXBOX FROGGY
Level 16 of 24

Another property you can apply to individual items is `align-self`. This property accepts the same values as `align-items` and its value for the specific item.

```
#pond {
  display: flex;
  align-items: flex-start;
}

.yellow {
  align-self:end;
}
```

Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings
Want to learn CSS grid? Play Grid Garden.



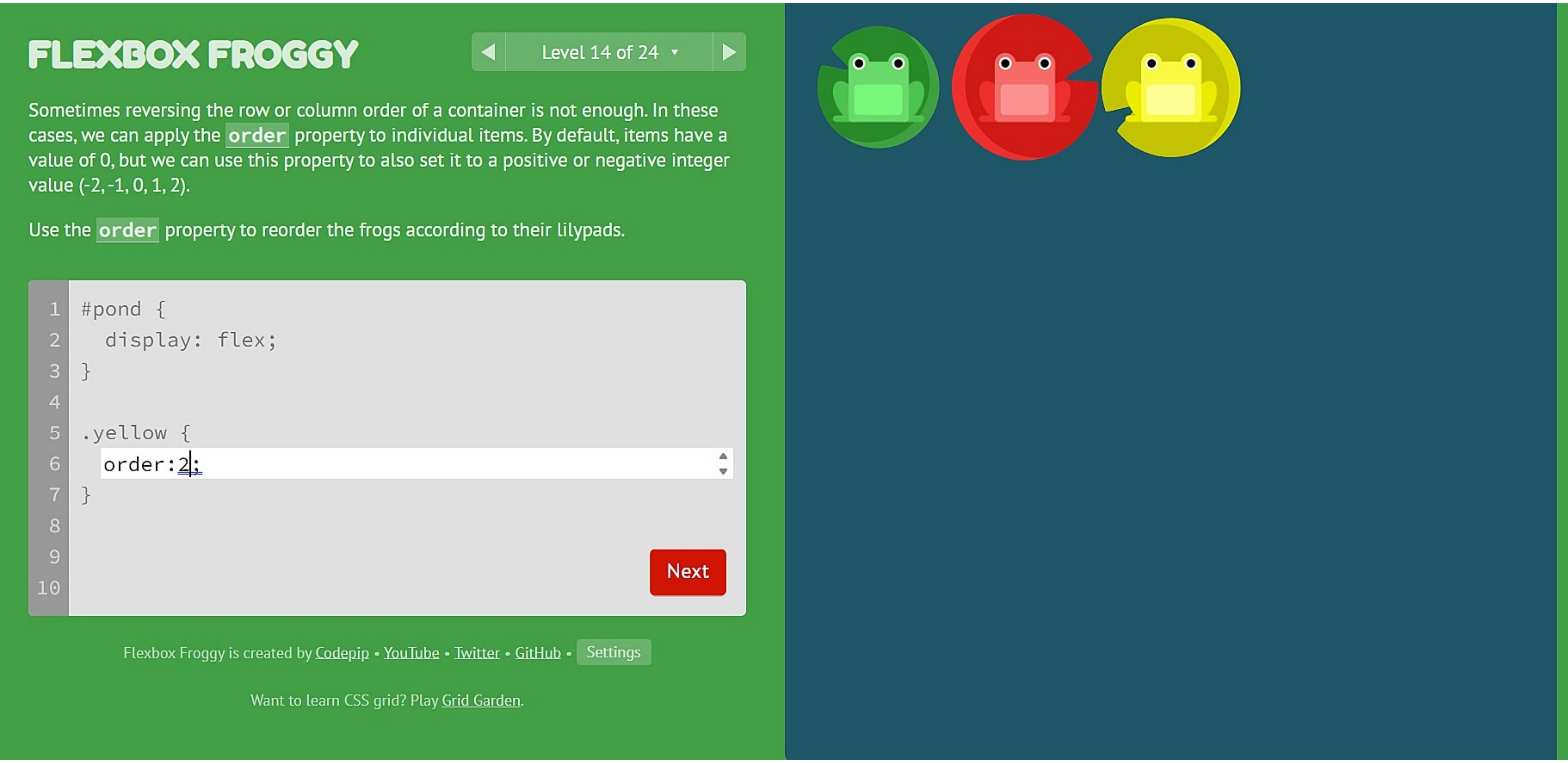
Use the `order` property to send the red frog to his lilypad.

```
#pond {
  display: flex;
}

.red {
  order:-3;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.



FLEXBOX FROGGY

Level 14 of 24

Sometimes reversing the row or column order of a container is not enough. In these cases, we can apply the `order` property to individual items. By default, items have a value of 0, but we can use this property to also set it to a positive or negative integer value (-2, -1, 0, 1, 2).

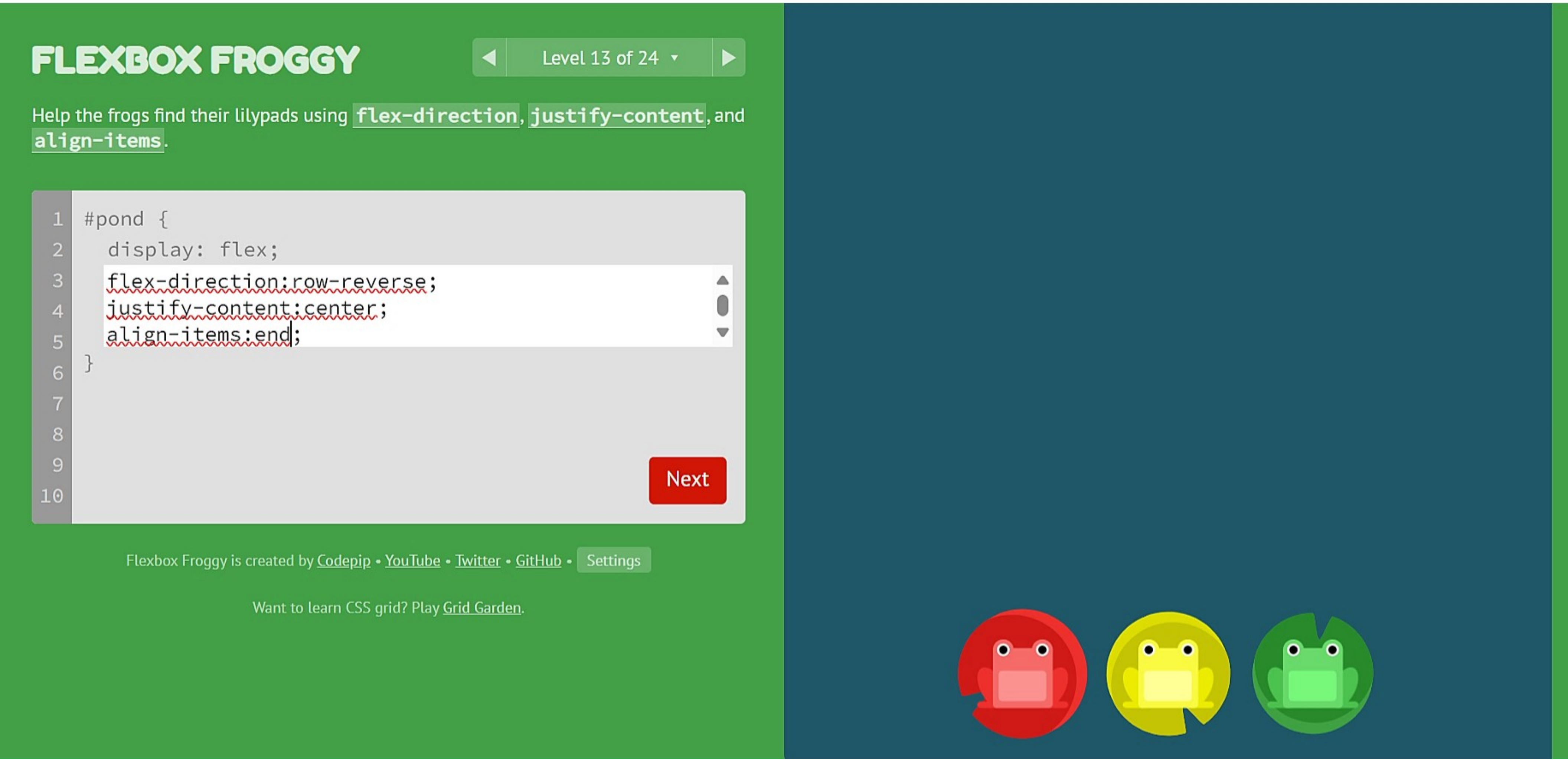
Use the `order` property to reorder the frogs according to their lilypads.

```
#pond {
  display: flex;
}

.yellow {
  order:2;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

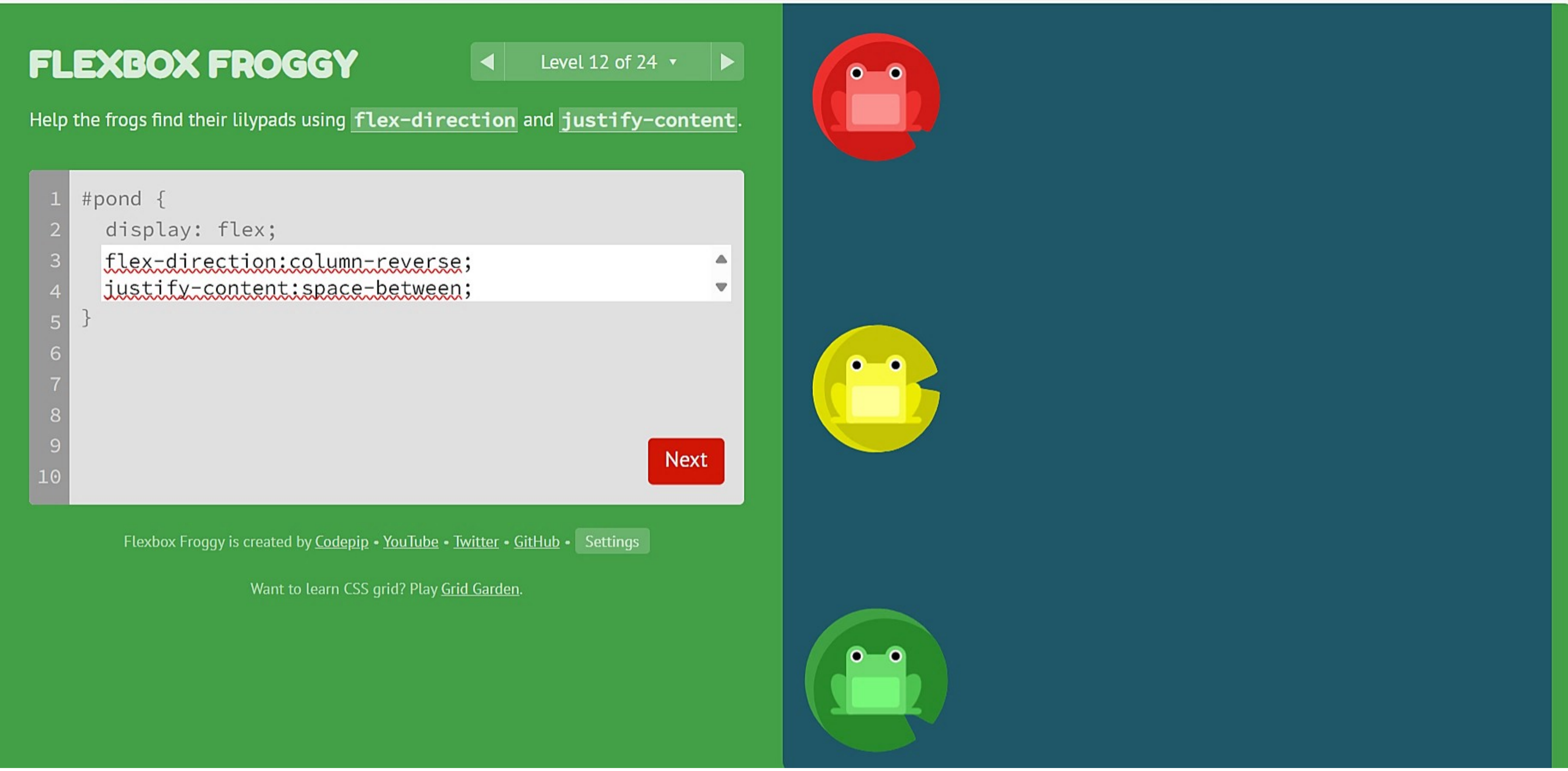
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Level 13 of 24

Help the frogs find their lilypads using `flex-direction`, `justify-content`, and `align-items`.

```
#pond {
  display: flex;
  flex-direction:row-reverse;
  justify-content:center;
  align-items:end;
}
```

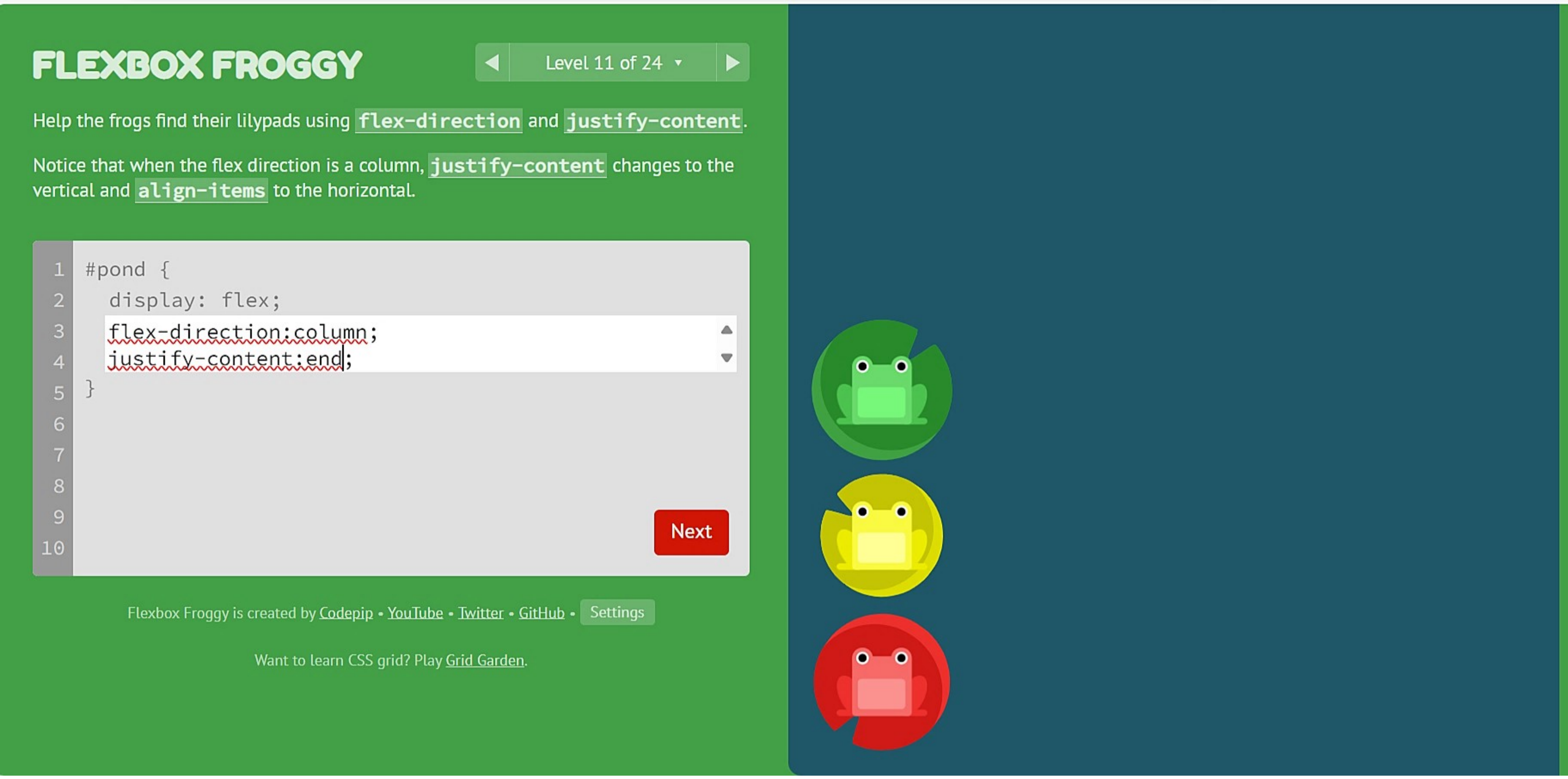
Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings
Want to learn CSS grid? Play Grid Garden.


FLEXBOX FROGGY
Level 12 of 24

Help the frogs find their lilypads using `flex-direction` and `justify-content`.

```
#pond {
  display: flex;
  flex-direction:column-reverse;
  justify-content:space-between;
}
```

Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings
Want to learn CSS grid? Play Grid Garden.



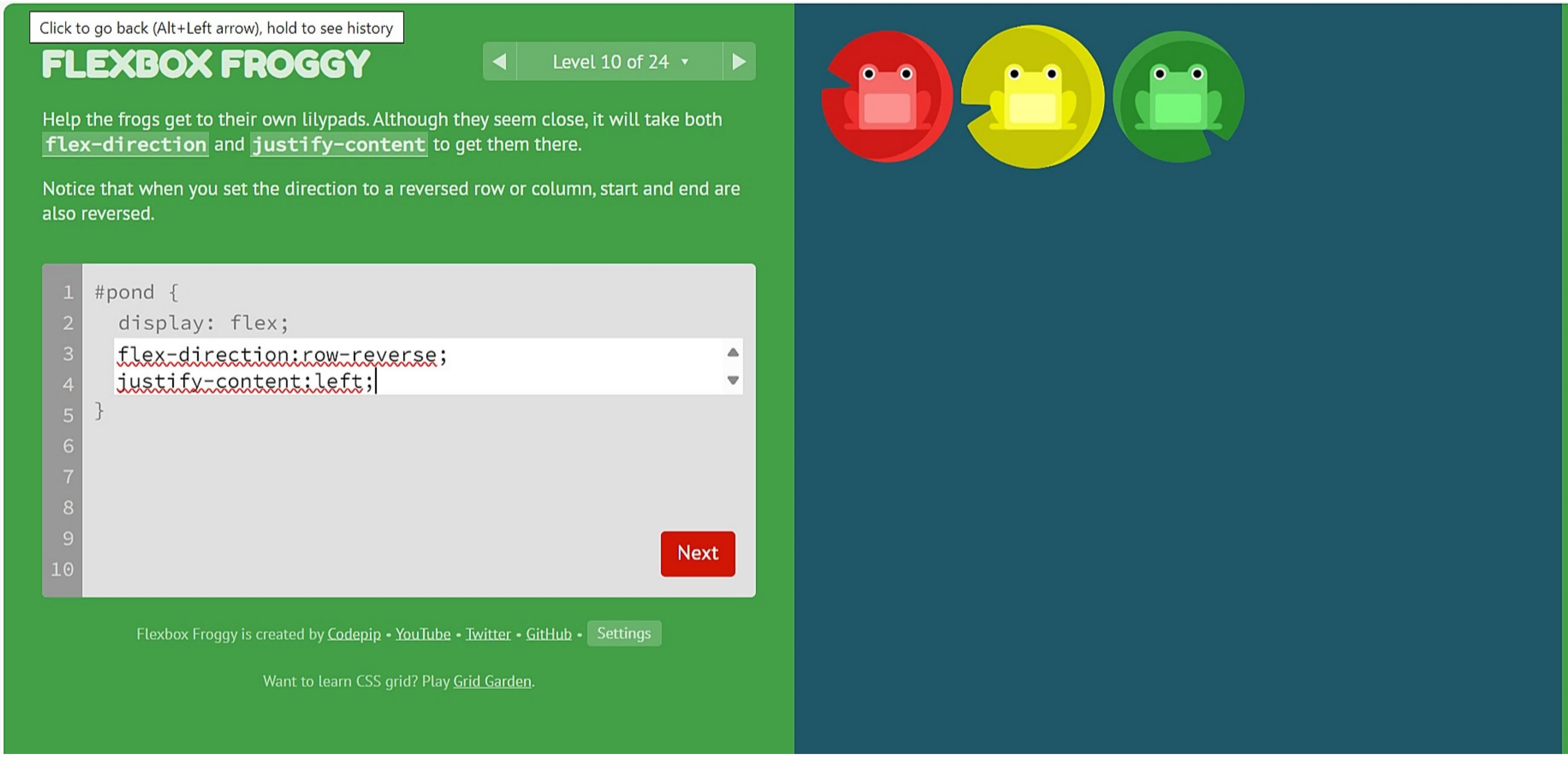
Help the frogs find their lilypads using `flex-direction` and `justify-content`.

Notice that when the flex direction is a column, `justify-content` changes to the vertical and `align-items` to the horizontal.

```
#pond {
  display: flex;
  flex-direction:column;
  justify-content:end;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub •

Want to learn CSS grid? Play Grid Garden.



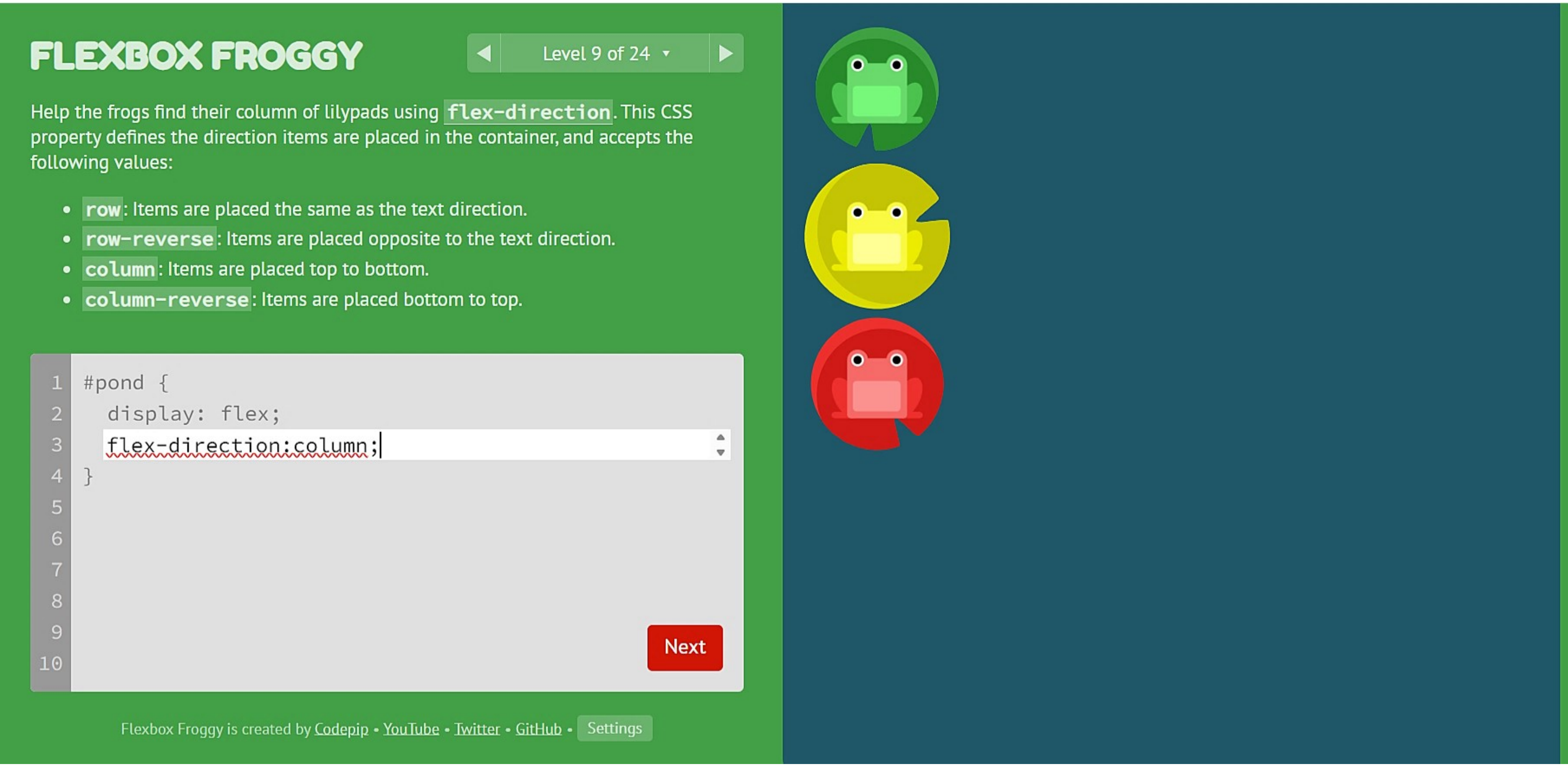
Help the frogs get to their own lilypads. Although they seem close, it will take both `flex-direction` and `justify-content` to get them there.

Notice that when you set the direction to a reversed row or column, start and end are also reversed.

```
#pond {
  display: flex;
  flex-direction:row-reverse;
  justify-content:left;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.

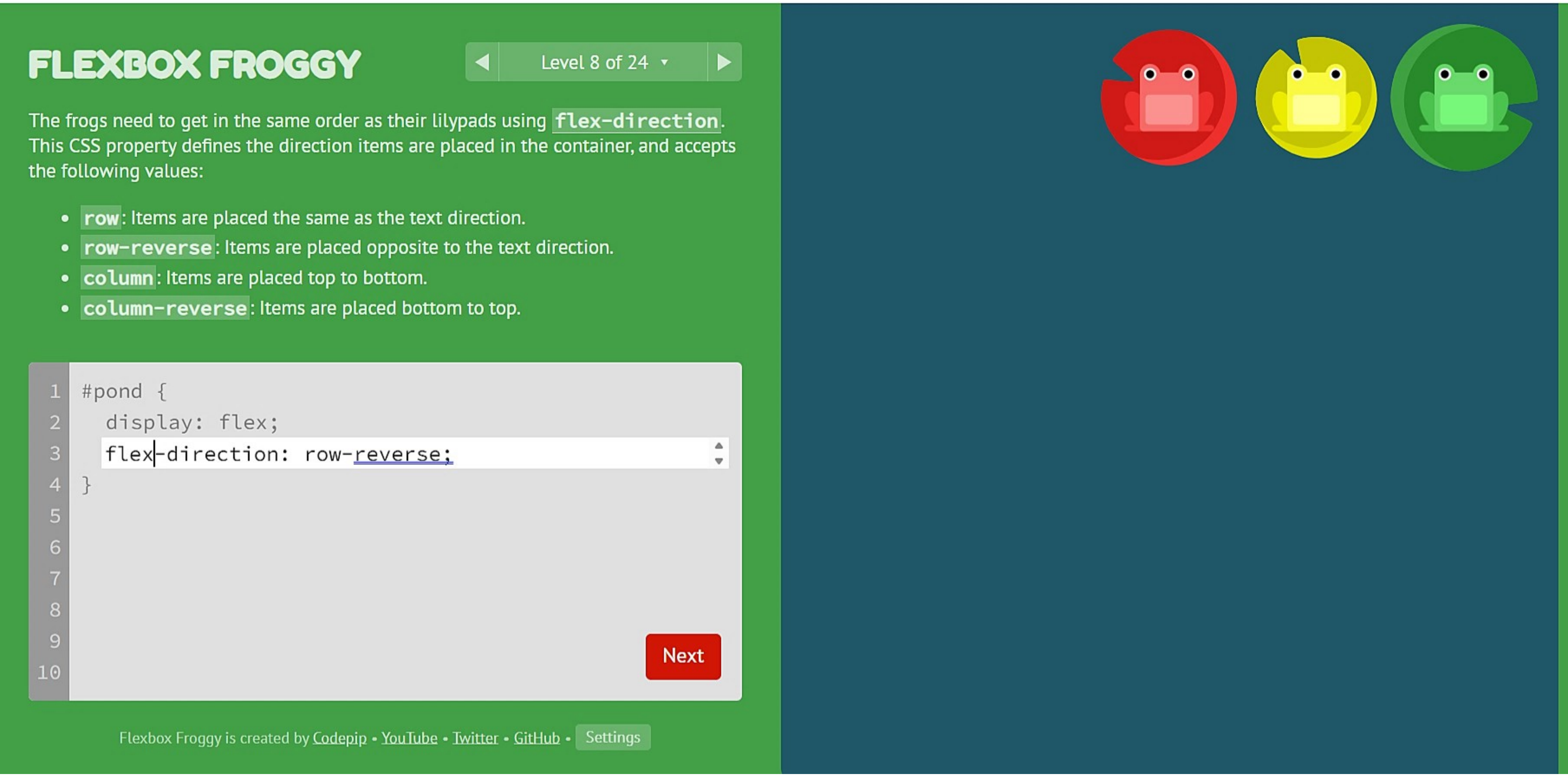


Help the frogs find their column of lilypads using `flex-direction`. This CSS property defines the direction items are placed in the container, and accepts the following values:

- `row`: Items are placed the same as the text direction.
- `row-reverse`: Items are placed opposite to the text direction.
- `column`: Items are placed top to bottom.
- `column-reverse`: Items are placed bottom to top.

```
#pond {
  display: flex;
  flex-direction:column;
}
```

Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

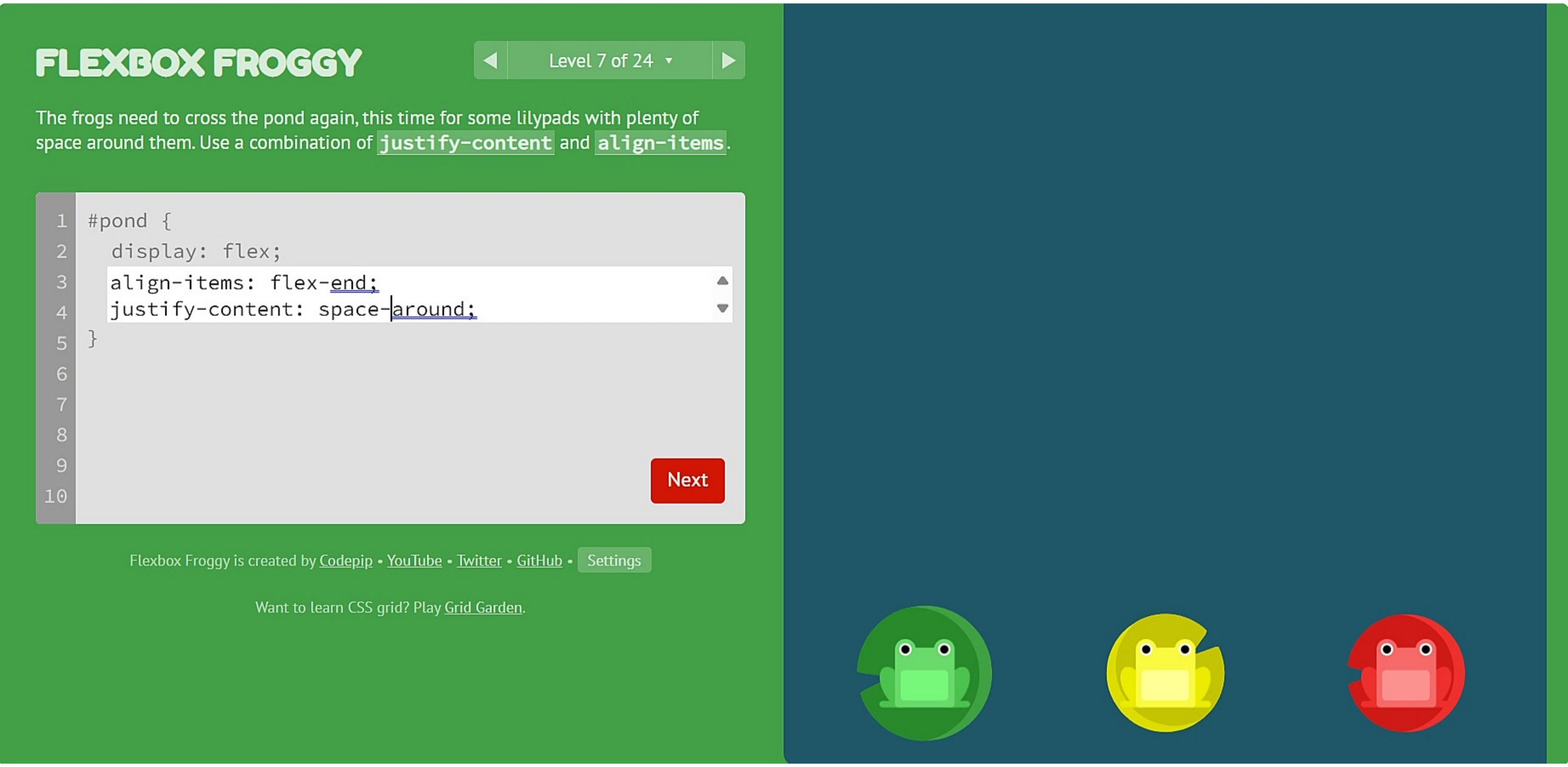


The frogs need to get in the same order as their lilypads using `flex-direction`. This CSS property defines the direction items are placed in the container, and accepts the following values:

- `row`: Items are placed the same as the text direction.
- `row-reverse`: Items are placed opposite to the text direction.
- `column`: Items are placed top to bottom.
- `column-reverse`: Items are placed bottom to top.

```
#pond {
  display: flex;
  flex-direction: row-reverse;
}
```

Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings


FLEXBOX FROGGY

Level 7 of 24

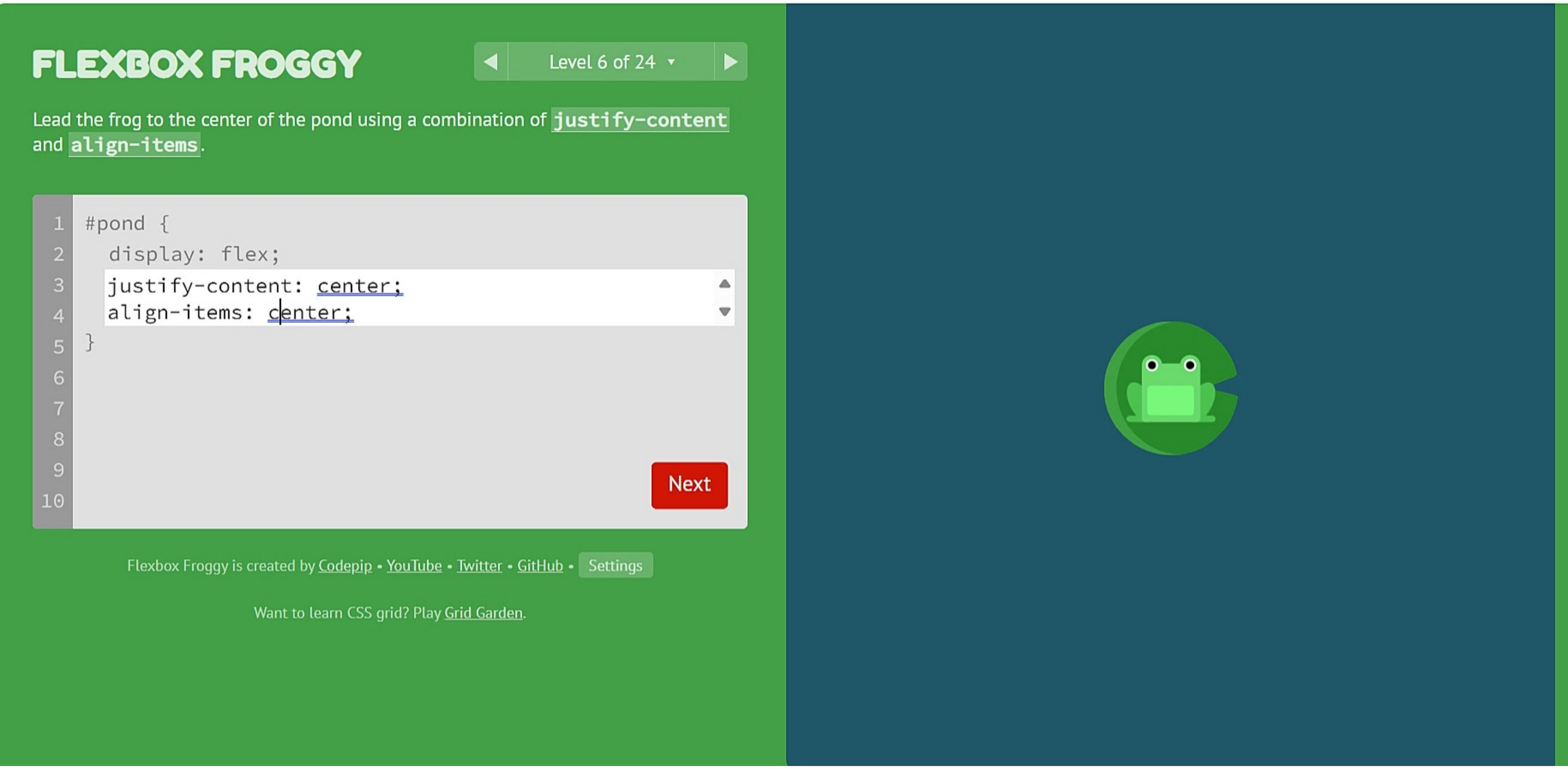
The frogs need to cross the pond again, this time for some lilypads with plenty of space around them. Use a combination of `justify-content` and `align-items`.

```
#pond {
  display: flex;
  align-items: flex-end;
  justify-content: space-around;
}
```

Next

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.

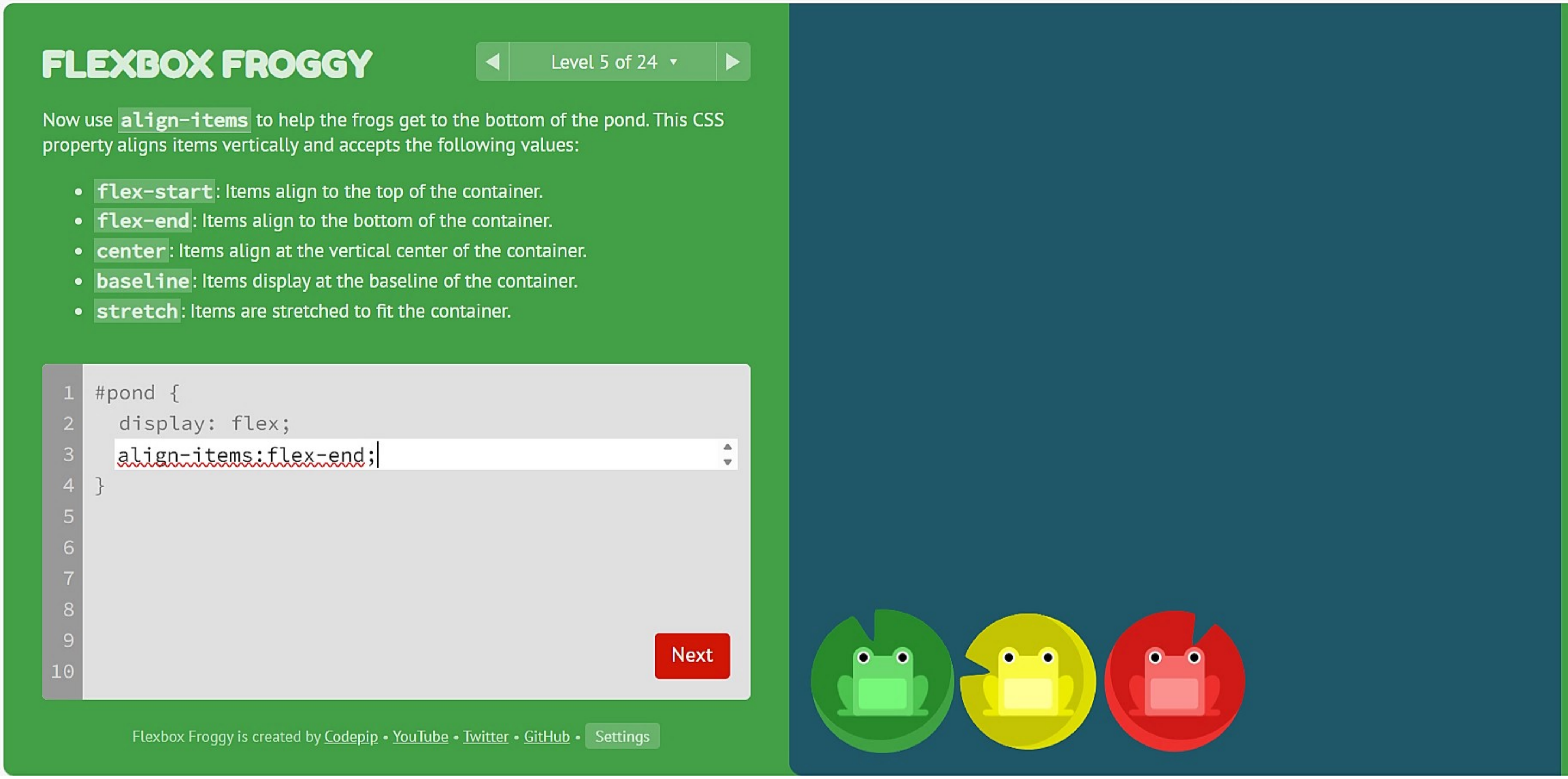


Lead the frog to the center of the pond using a combination of `justify-content` and `align-items`.

```
#pond {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub •

Want to learn CSS grid? Play Grid Garden.

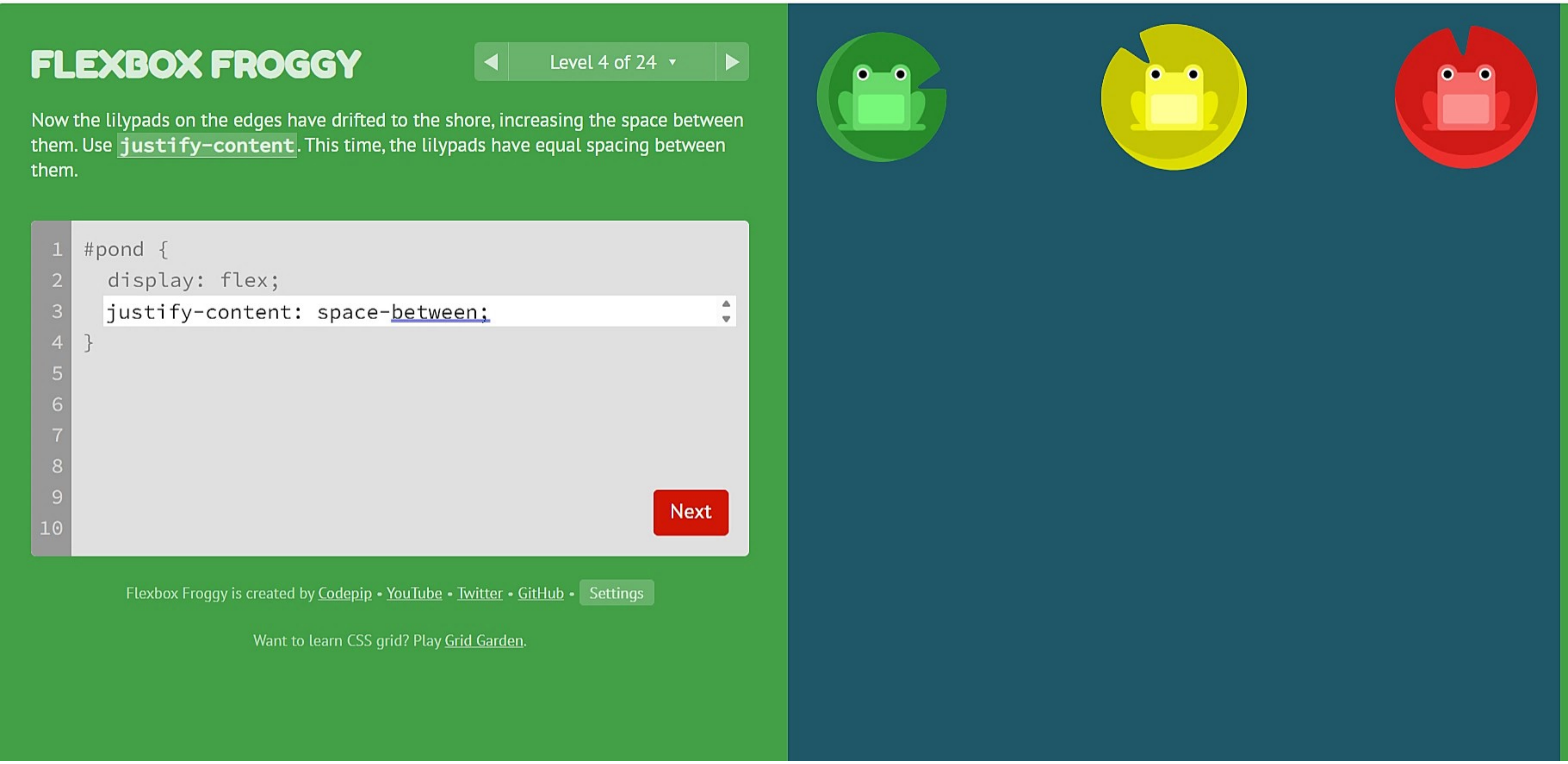


Now use `align-items` to help the frogs get to the bottom of the pond. This CSS property aligns items vertically and accepts the following values:

- `flex-start`: Items align to the top of the container.
- `flex-end`: Items align to the bottom of the container.
- `center`: Items align at the vertical center of the container.
- `baseline`: Items display at the baseline of the container.
- `stretch`: Items are stretched to fit the container.

```
#pond {
  display: flex;
  align-items:flex-end;
}
```

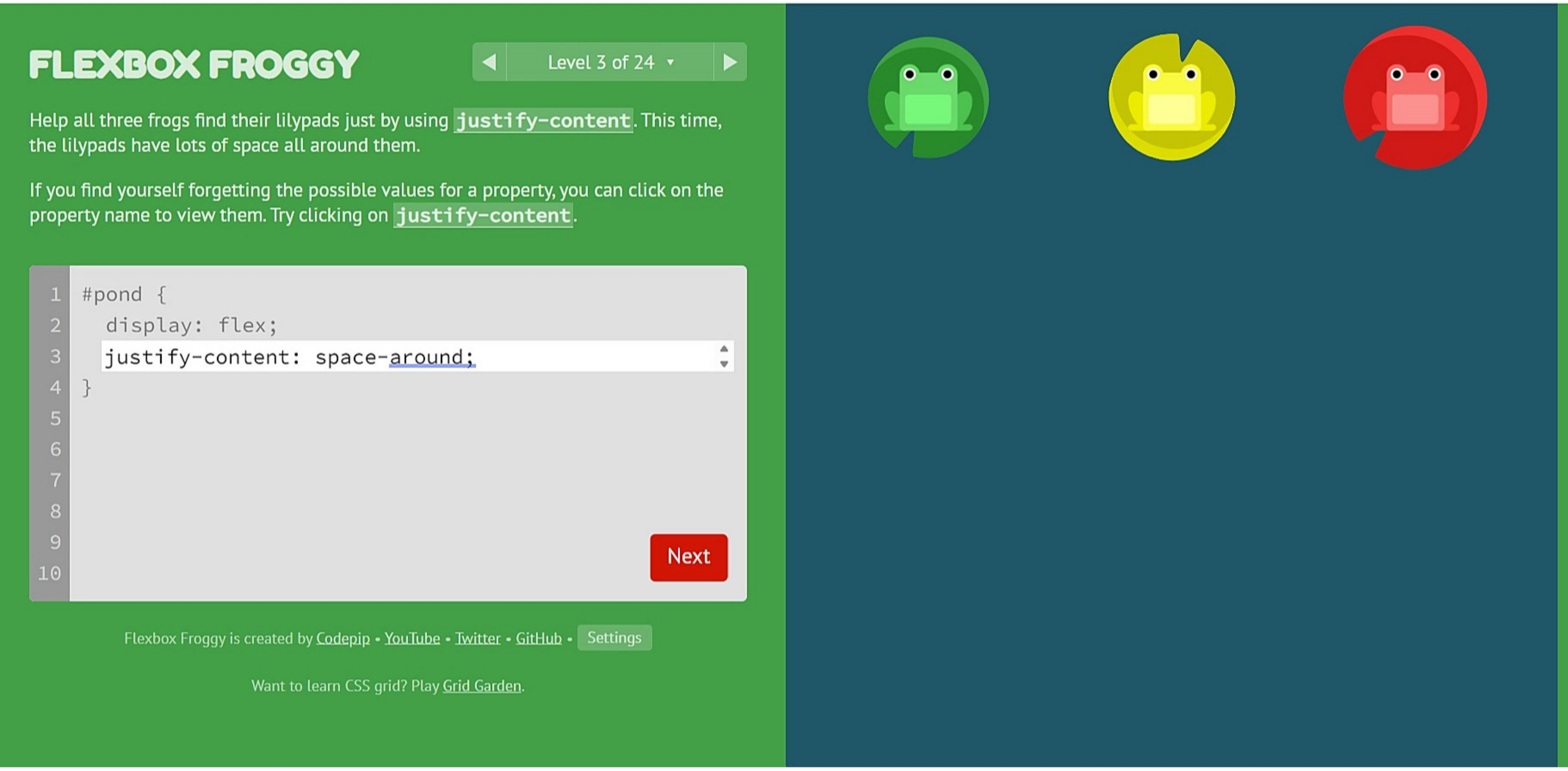
Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings


FLEXBOX FROGGY
Level 4 of 24

Now the lilypads on the edges have drifted to the shore, increasing the space between them. Use `justify-content`. This time, the lilypads have equal spacing between them.

```
#pond {
  display: flex;
  justify-content: space-between;
}
```

Next
Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings
Want to learn CSS grid? Play Grid Garden.



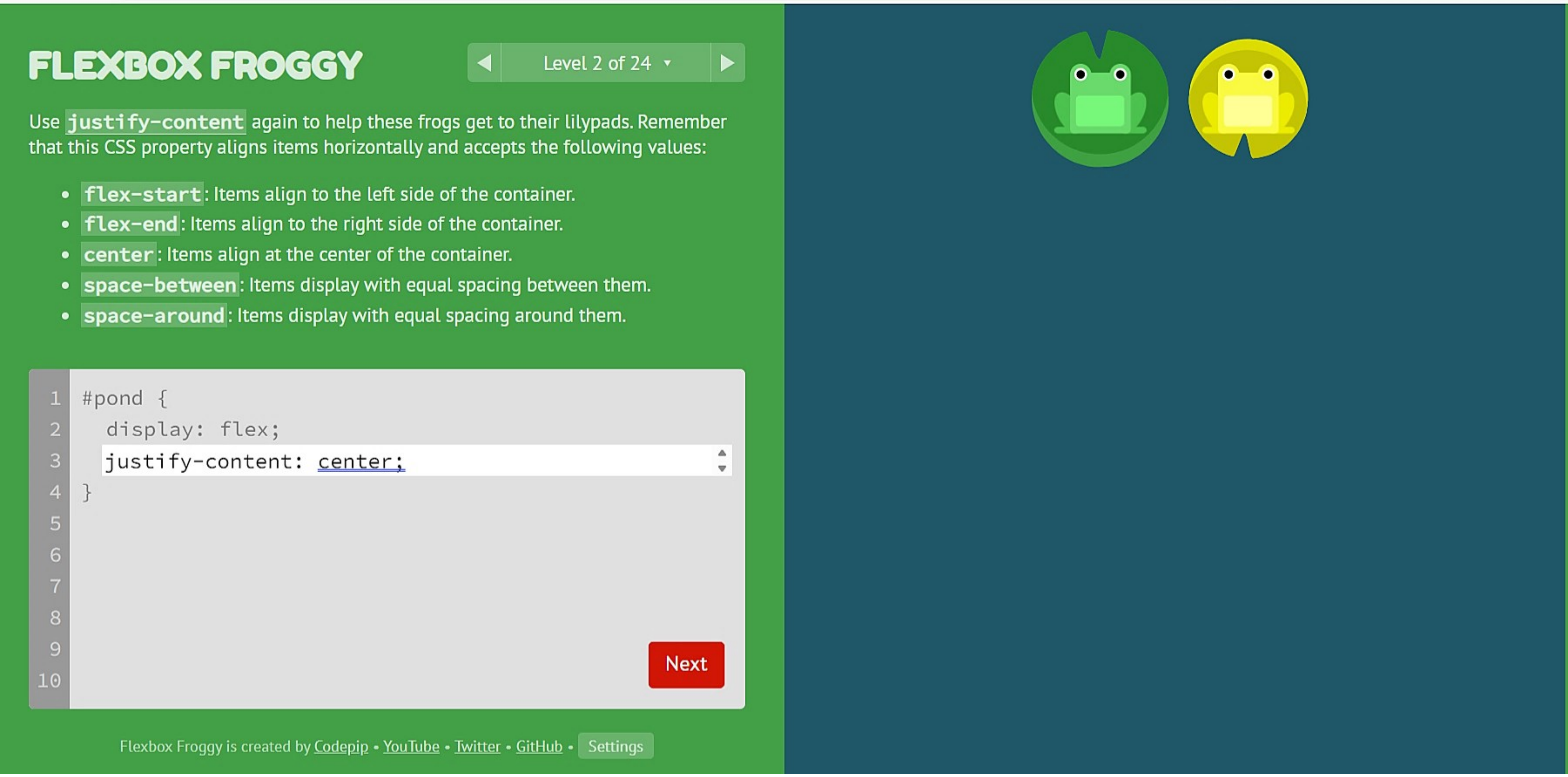
Help all three frogs find their lilypads just by using `justify-content`. This time, the lilypads have lots of space all around them.

If you find yourself forgetting the possible values for a property, you can click on the property name to view them. Try clicking on `justify-content`.

```
#pond {
  display: flex;
  justify-content: space-around;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings

Want to learn CSS grid? Play Grid Garden.



FLEXBOX FROGGY

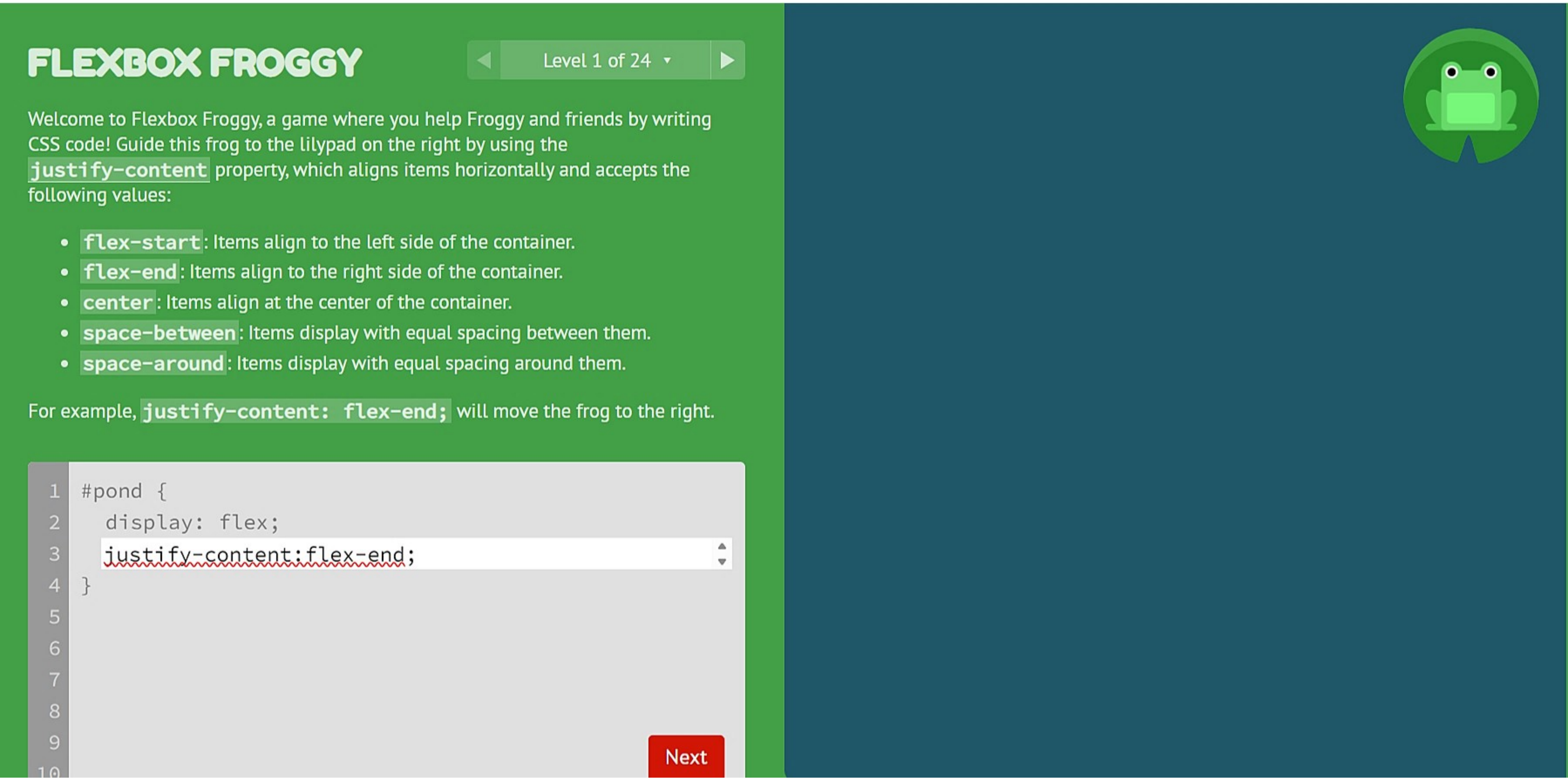
Level 2 of 24

Use `justify-content` again to help these frogs get to their lilypads. Remember that this CSS property aligns items horizontally and accepts the following values:

- `flex-start`: Items align to the left side of the container.
- `flex-end`: Items align to the right side of the container.
- `center`: Items align at the center of the container.
- `space-between`: Items display with equal spacing between them.
- `space-around`: Items display with equal spacing around them.

```
#pond {
  display: flex;
  justify-content: center;
}
```

Flexbox Froggy is created by Codepip • YouTube • Twitter • GitHub • Settings



Welcome to Flexbox Froggy, a game where you help Froggy and friends by writing CSS code! Guide this frog to the lilypad on the right by using the `justify-content` property, which aligns items horizontally and accepts the following values:

- `flex-start`: Items align to the left side of the container.
- `flex-end`: Items align to the right side of the container.
- `center`: Items align at the center of the container.
- `space-between`: Items display with equal spacing between them.
- `space-around`: Items display with equal spacing around them.

For example, `justify-content: flex-end;` will move the frog to the right.

```
#pond {
  display: flex;
  justify-content:flex-end;
}
```