Your grade: 100%
Your latest: 100%•Your highest: 100%•To pass you need at least 80%. We keep your highest score.

Next item

1.What is a `Theme` composed of? 1 / 1 point

- A collection of views
- A collection of attributes
- A collection of composables

Correct
Correct! `Theme` is represented by attributes such as height and color.

2.Which of the following hexadecimal values will produce the white color? 1 / 1 point

- `0xFF000000`
- `0x00000000`
- `0xFFFFFFFF`
- `0x00FFFFFF`

Correct
Correct! A value of FF for the first two characters means fully opaque. Setting FF for the remaining six characters means full color.

3.True or False: The main difference between `dp` (density independent pixel) and `sp` (scalable pixel) is that it takes the text size setting of the user device into consideration when deciding what text size should be displayed. 1 / 1 point

- True
- False

Correct
Correct! The user may modify this setting in the Android system preferences, and the text size is scaled accordingly in the application.

4.Which of the following correctly creates a text composable with underlined text? 1 / 1 point

```
Text(
    style = TextDecoration.Underline
)
```

```
Text(
    style = TextStyle(
        text = “Hello World!”,
        fontSize = 25.sp,
        decoration = TextDecoration.Underline
 )
)
```

`Text(`

```
    style = TextStyle(
        text = “Hello World!”,
        fontSize = 25.sp,
        textDecoration = underline
    )
)
```

```
Text(
    style = TextStyle(
        text = “Hello World!”,
        fontSize = 25.sp,
        textDecoration = TextDecoration.Underline
    )
)
```

Correct
Correct! The text-decoration property can be used to apply underline text. The `TextDecoration` combine function enables the application of multiple decorations.

5.Which of the following correctly adds `Text` composable as the content for a `Surface`? 1 / 1 point

```
Surface(
{
    Text(
            text = “Hello World”,
            fontSize = 30.sp
            )
}
)
```

```
Text(
    modifier = Modifier
        .fillMaxSize()
) {
    Surface(
            text = “Hello World”,
            fontSize = 30.sp
            )
}
```

```
Surface(
    modifier = Modifier
        .fillMaxSize()
) {
    Text(
            text = “Hello World”,
            fontSize = 30.sp
            )
}
```

```
Surface(
    modifier = Modifier
        .fillMaxSize()
    text = “Hello World”,
    fontSize = 30.sp
)
}
```

Correct
Correct! Like layout composables, the `Surface` composable can have content such as the `Text` composable..

6.Which of the following is an optional modifier that can be used to change the color of text within a `Text` composable? 1 / 1 point

- `fontColor`
- `color`
- `textColor`
- `text`

Correct
Correct! A Text composable accepts over 15 optional parameters including `color` for styling the text.

7.Is it possible to change the shape of `Surface` corners? 1 / 1 point

- Yes
- No

Correct
Correct! By default, the `Surface` has a rectangular shape. You can modify the shape of the `Surface` by adding the shape parameter.

8.Which of the following is one of the most common APIs provided by Jetpack Compose to enable various animations? 1 / 1 point

- appearance API
- animated visibility API
- animation API
- visibility API

Correct
Correct! The animated visibility API animates the appearance and disappearance of its content.

9.Which of the following will make the text “Hello World” slide up to disappear and slide down to disappear? 1 / 1 point

```
var visible by remember {
    mutableStateOf(true)
}
Column {
    if  (visible)  {
        Text(text = “Hello World”)
    }
    Button(onClick = Text(“Button  “))
```

```
var visible by remember {
    mutableStateOf(true)
}
Column {
    if  (visible)  {

        Text(text = “Hello World”)
    }
    Button(onClick = {  visible = !visible  })  {
        Text(“Button”)
}
}
```

```
var visible by remember {
    mutableStateOf(true)
}
Column  {
    AnimatedVisibility(visible)  {
        Text(text = “Hello World”)
    }
    Button(onClick = {  visible = !visible  })  {
        Text(“Button  “)
    }
}
```

```
var visible by remember {
    mutableStateOf(true)
}
Column  {
    AnimatedVisibility(visible) {
        Text(text = “Hello World”)
    }
    Button(onClick = Text(“Button“))
```