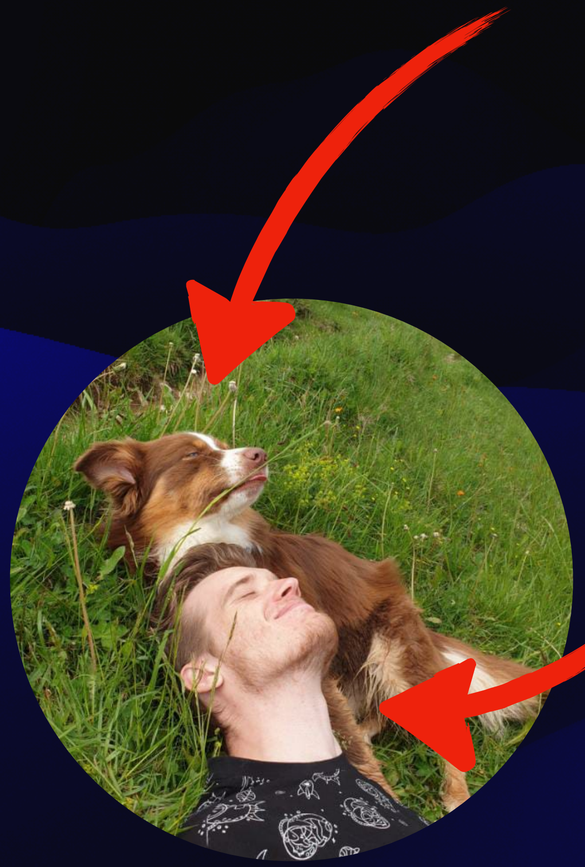
Inside Jaspr: Building a Dart web framework from scratch

Kilian Schulte
@schultek_dev

Building websites sucks!

Jaspr is a framework
to build websites
in a familiar way

Jasper



Kilian Schulte

@schultek

@schultek_dev

```
class FooComponent extends StatelessComponent {
  const FooComponent({super.key});

  @override
  Iterable<Component> build(BuildContext context) sync* {
    yield div([
      p([text('Hello World')]),
    ]);
    yield img(src: '/images/logo.png');
  }
}
```

```dart
class FooComponent extends StatelessComponent {
  const FooComponent({super.key});

  @override
  Iterable<Component> build(BuildContext context) sync* {
    yield div([
      p([text('Hello World')]),
    ]);
    yield img(src: '/images/logo.png');
  }
}
```

```html
<div>
    <p>Hello World</p>
</div>
<img src="/images/logo.png" />
```

```
class FooComponent extends StatelessComponent {
  const FooComponent({super.key});

  @override
  Iterable<Component> build(BuildContext context) sync* {
    yield div([
      p([text('Hello World')]),
    ]);
    yield img(src: '/images/logo.png');
  }
}
```

```
Node a = document.createElement('p');
a.text = 'Hello World';
Node b = document.createElement('div');
b.append(a);
Node c = document.createElement('img');
c.setAttribute('src', '/images/logo.png');
```

```
<div>
    <p>Hello World</p>
</div>
<img src="/images/logo.png" />
```

Flutters Rendering System

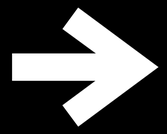
Widget

FooWidget

Holds config for a piece of UI

Has a public API

Frequently rebuilds

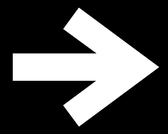


Element

FooElement

Represents an actual piece of the UI

Rarely rebuilds



RenderObject

RenderFoo

Renders to the screen

Rarely rebuilds

Jasprs Rendering System

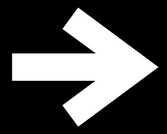
Component

FooComponent

Holds config for a piece of UI

Has a public API

Frequently rebuilds

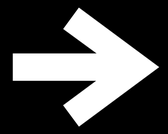


Element

FooElement

Represents an actual piece of the UI

Rarely rebuilds



DOM Node

RenderFoo

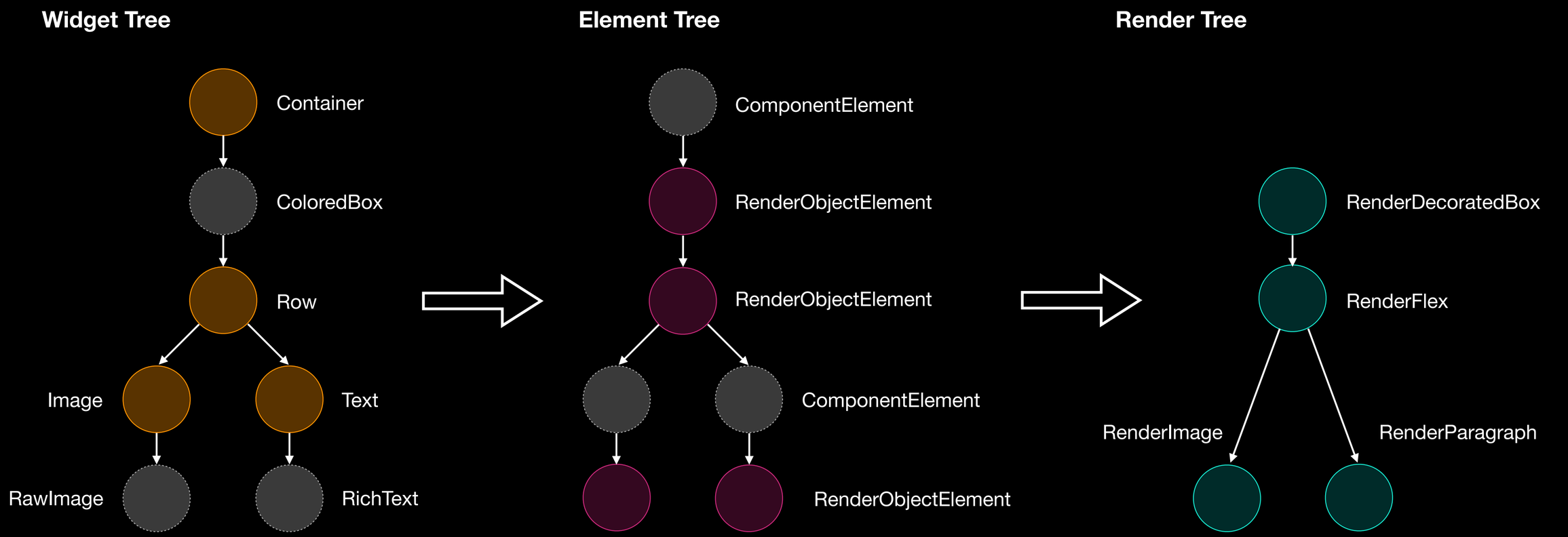
Just plain DOM nodes

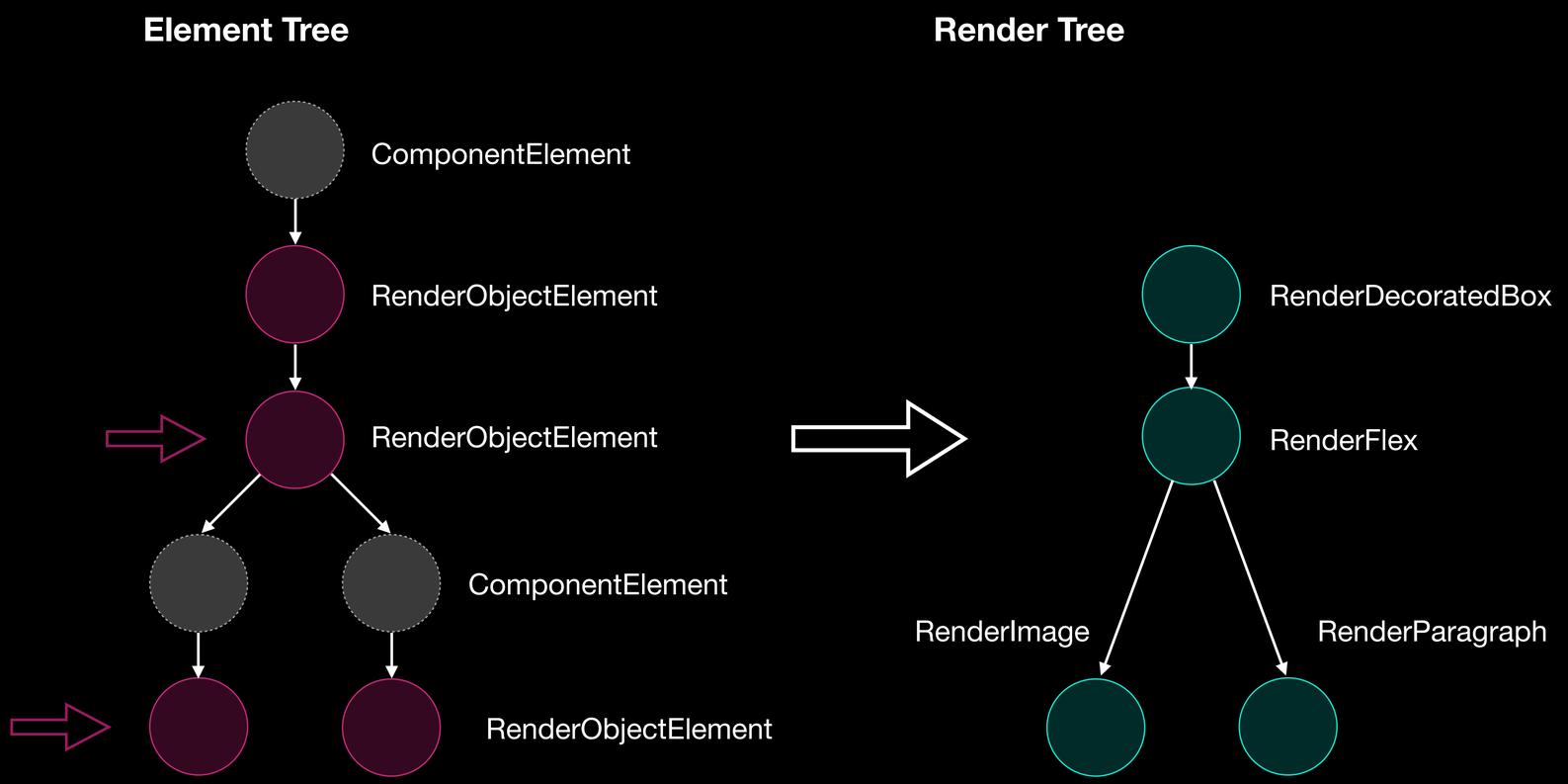
Rarely rebuilds


I MADE THIS
YOU MADE THIS?
Widget System
Widget System
Widget System
+786 −1,996
I MADE THIS
Component System

How does rendering actually work?

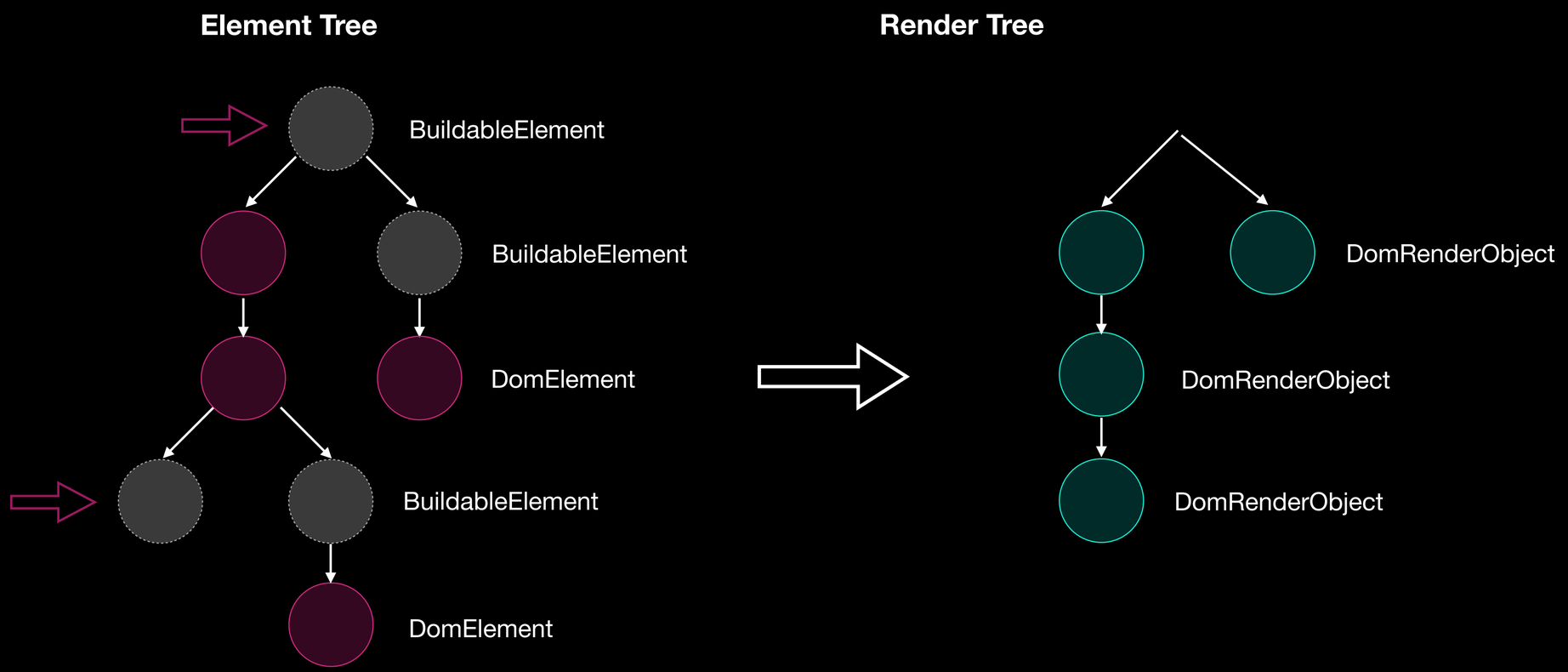

294
295 @protected
296 - Widget build(BuildContext context);
297
177
178 @protected
179 + Iterable<Component> build(BuildContext context);
180


Widget Tree
Container
ColoredBox
Row
Image
Text
RawImage
RichText
Element Tree
ComponentElement
RenderObjectElement
RenderObjectElement
ComponentElement
RenderObjectElement
Render Tree
RenderDecoratedBox
RenderFlex
RenderImage
RenderParagraph



Tree Invariants:

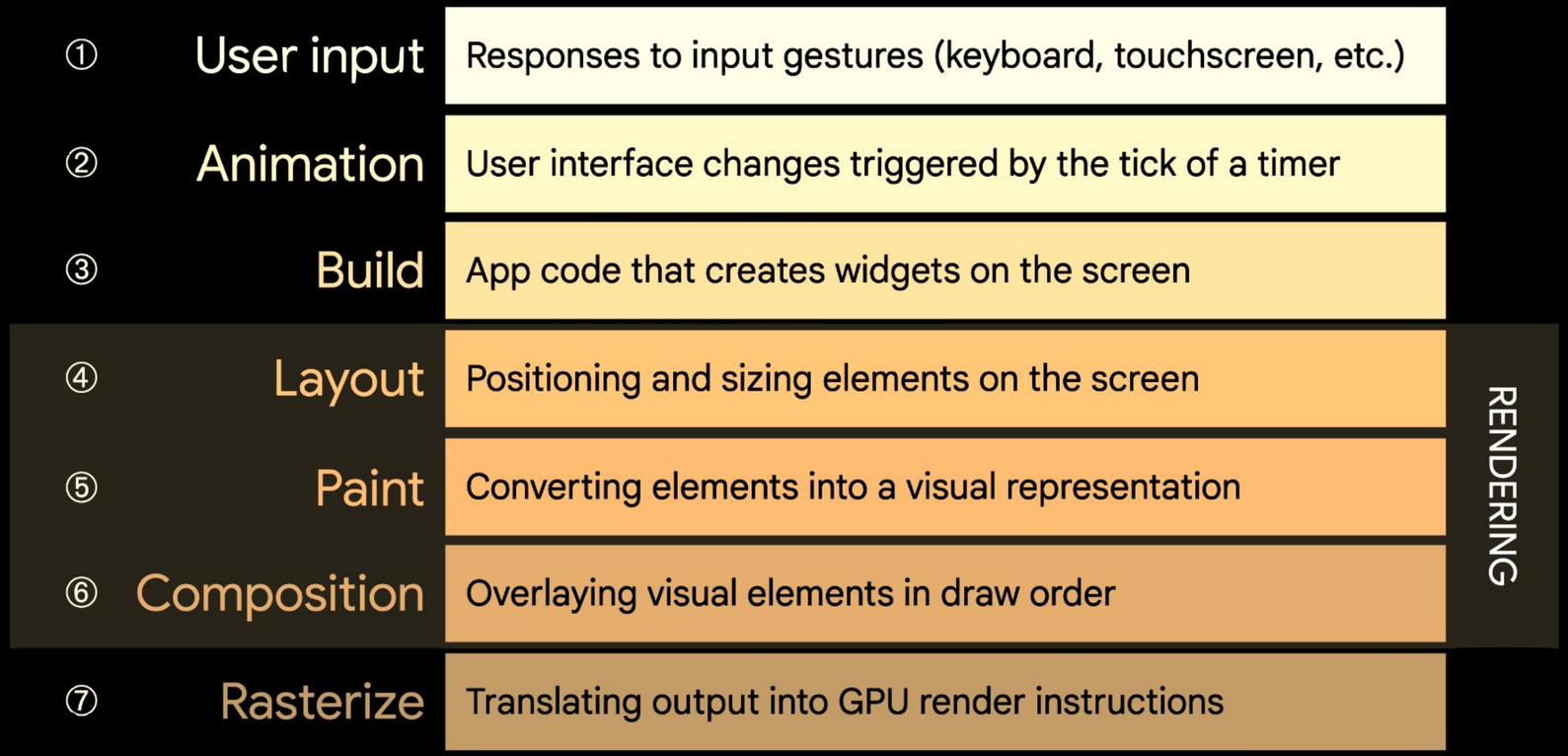
1. Only RenderObjectElements can have multiple children.
2. The leaf node is always a RenderObjectElement.



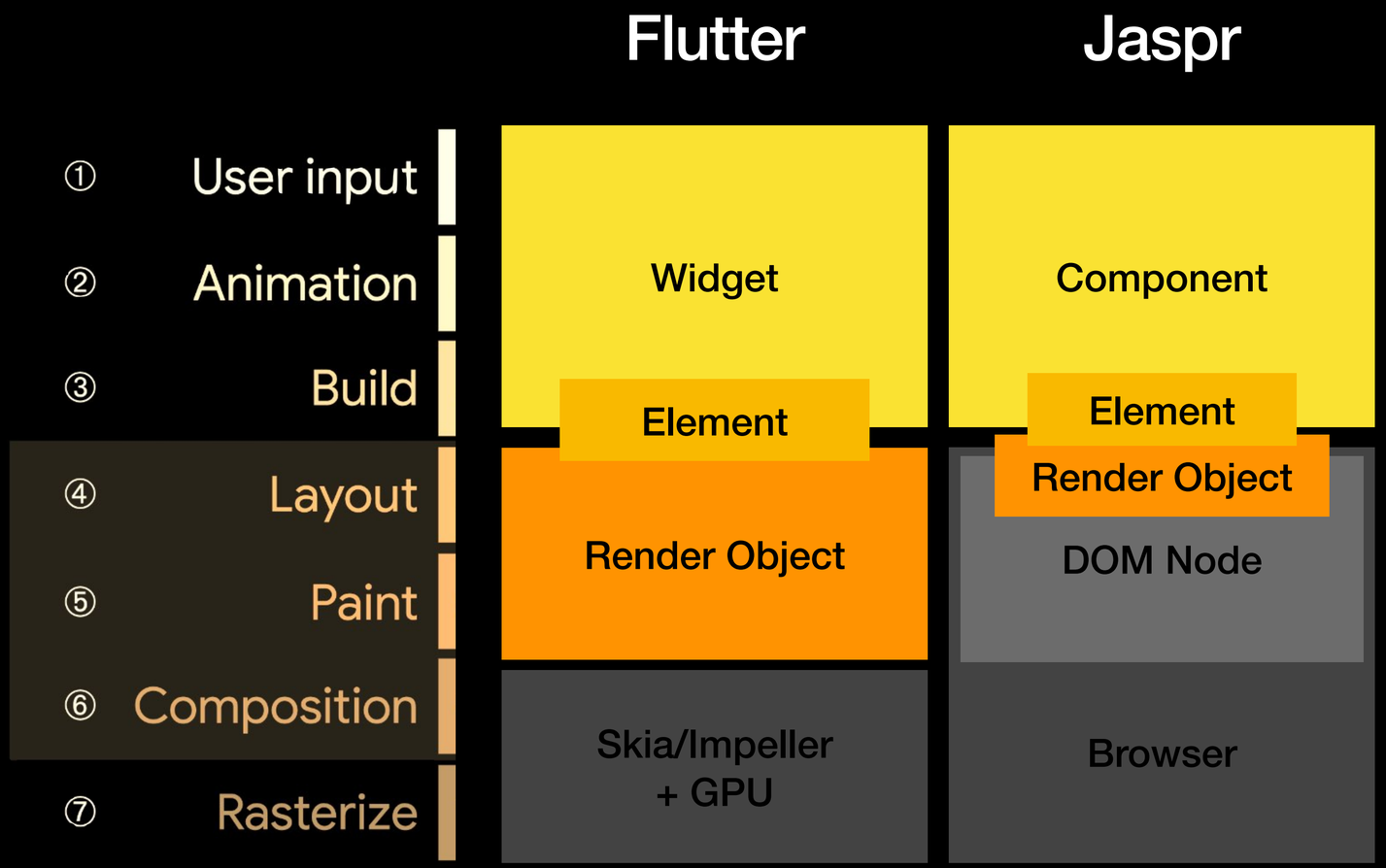
Tree Invariants:

- ✗ Any element can have multiple children.
- ✗ Not every branch has a RenderObjectElement

Rendering Pipeline



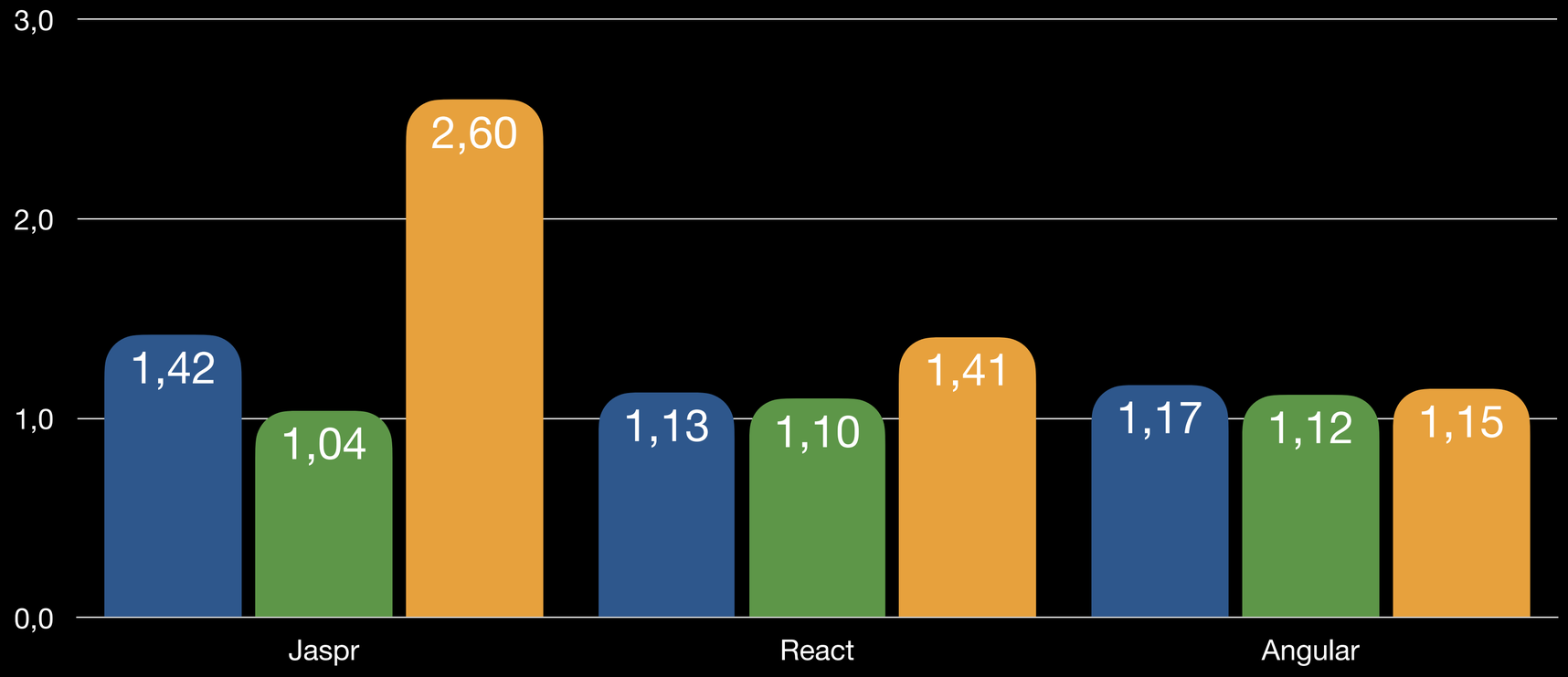
Rendering Pipeline



```dart
import 'dart:html' show Node;

class DomRenderObject {
  Node? node;

  /* ... */

  void updateElement(String tag, String? id, String? classes, Map<String, String>? styles,
      Map<String, String>? attributes, Map<String, EventCallback>? events) {

    if (node == null || node.tag != tag) {
      node = document.createElement(tag);
    }

    node.setAttribute('id', id);
    node.setAttribute('class', classes);
    node.setAttribute('style', styles?.entries.map((e) => '${e.key}: ${e.value}').join('; '));

    for (var attr in attributes.entries) {
      node.setAttribute(attr.key, attr.value);
    }

    /* ... */
  }

  /* ... */
}
```

Performance?

https://jaspr-benchmarks.web.app



Average slowness factors across all benchmarks (lower is better)

1.0 == always fastest, 1.5 == 50% slower as fastest

General Performance:

~20% slower as R / A

Time spent with rendering:

~5% faster than R / A

Time spent with executing javascript:

~45% / ~55% slower as R / A

Going beyond what Flutter can do!

```
runApp( FooComponent );
```

Jaspr

Rendering

on the
client

DOM instructions

Browser

HTML Elements

```
runApp( FooComponent );
```

Jaspr

Pre-rendering

on the
server

HTML Markup
(as a String)

Rendering

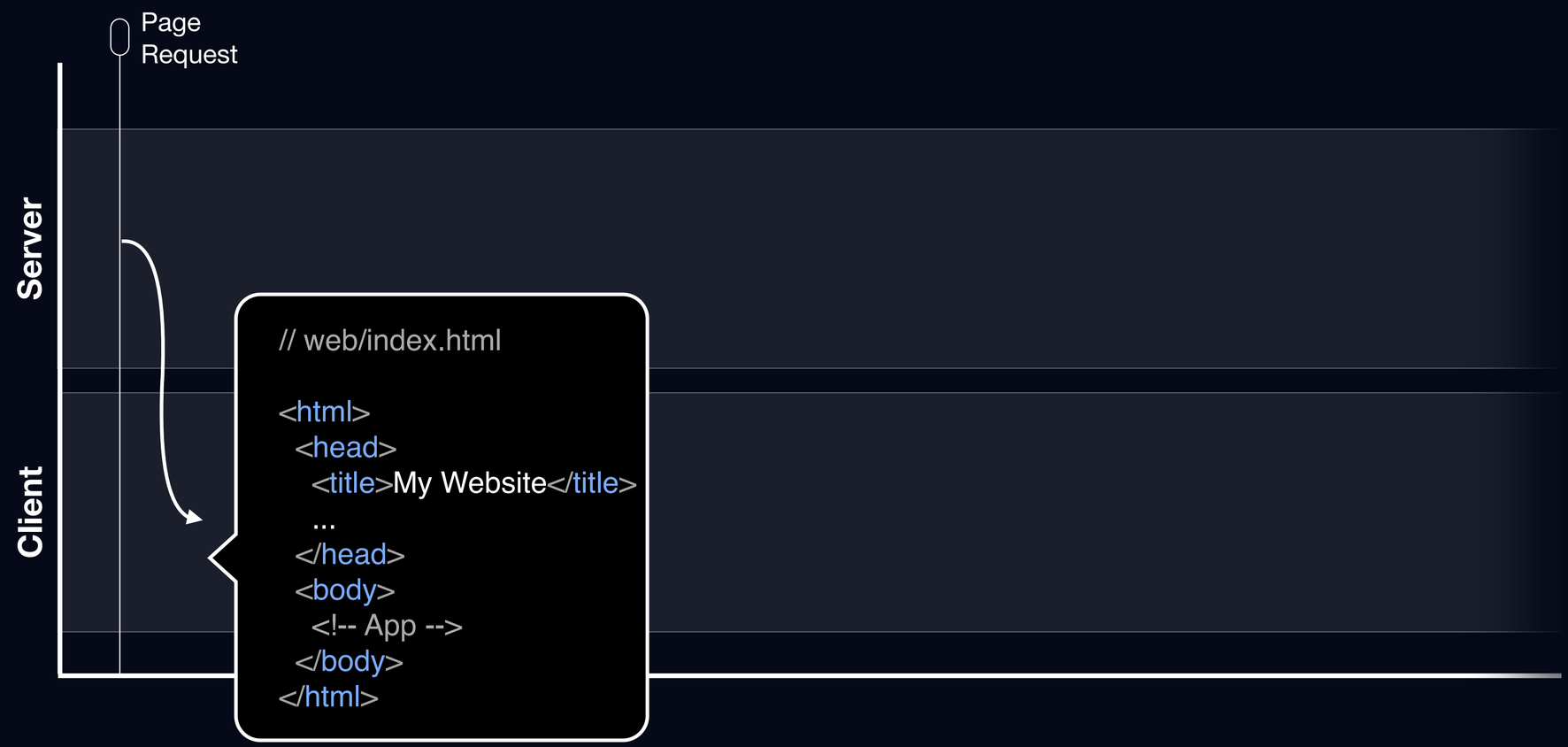
on the
client

DOM instructions

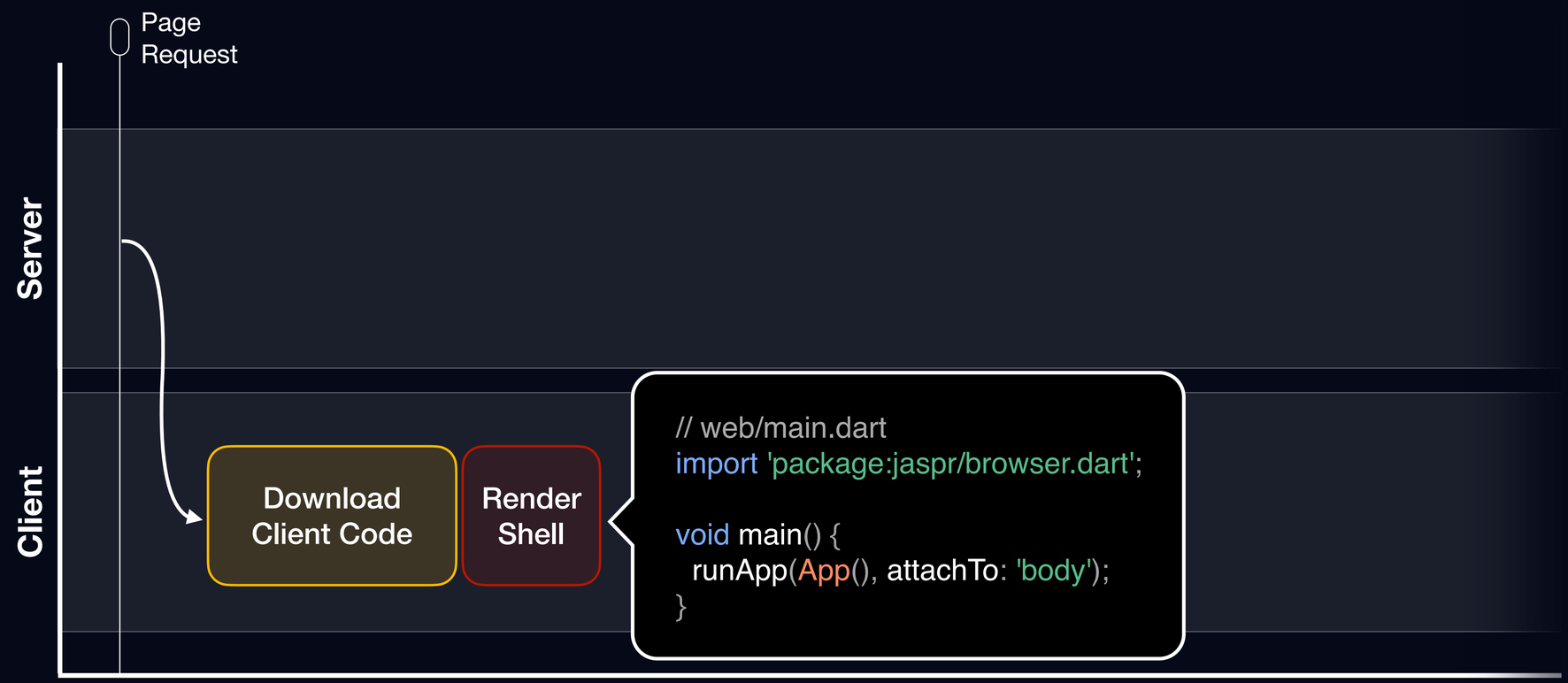
Browser

HTML Elements

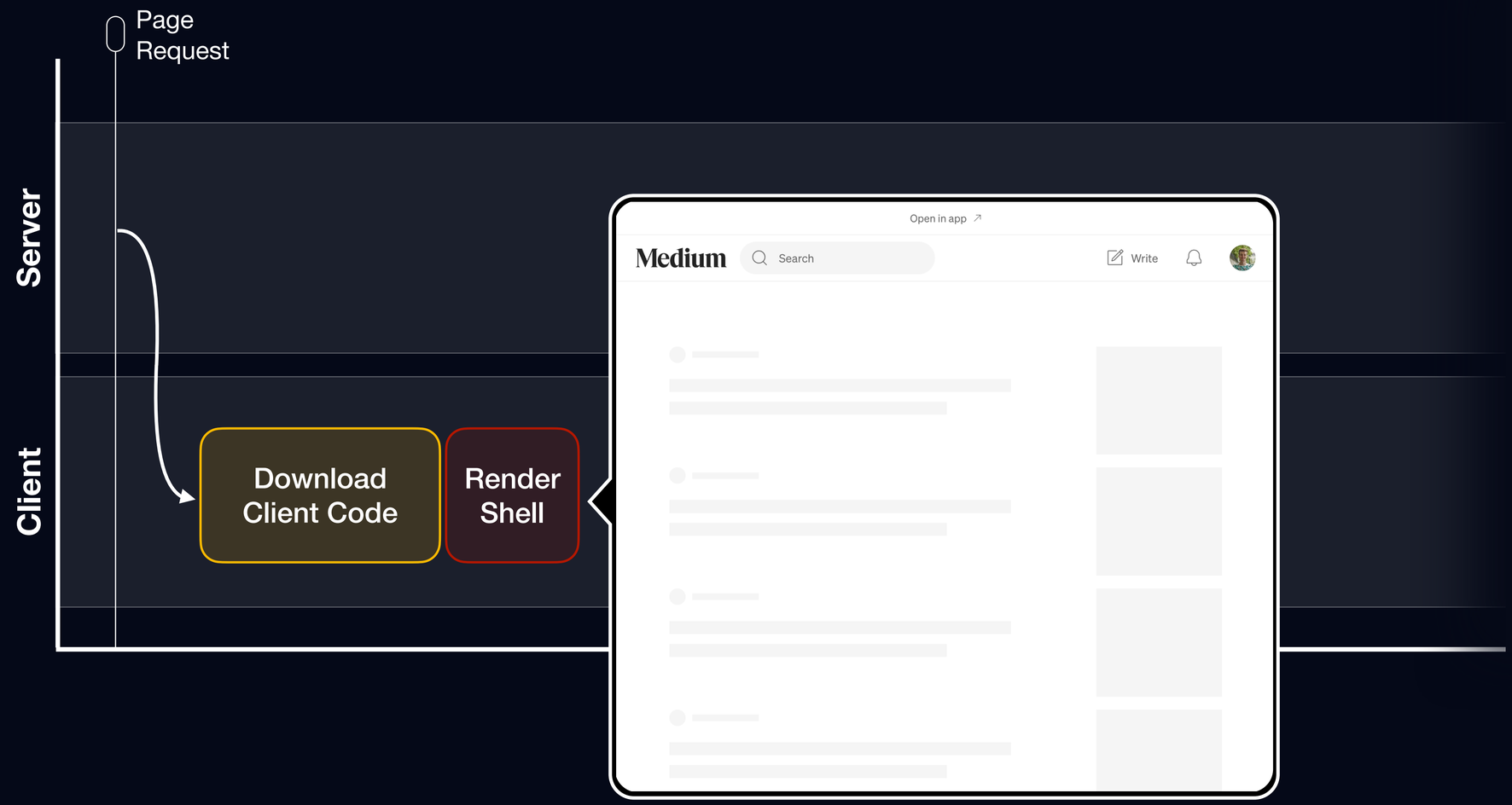
Client Side Rendering



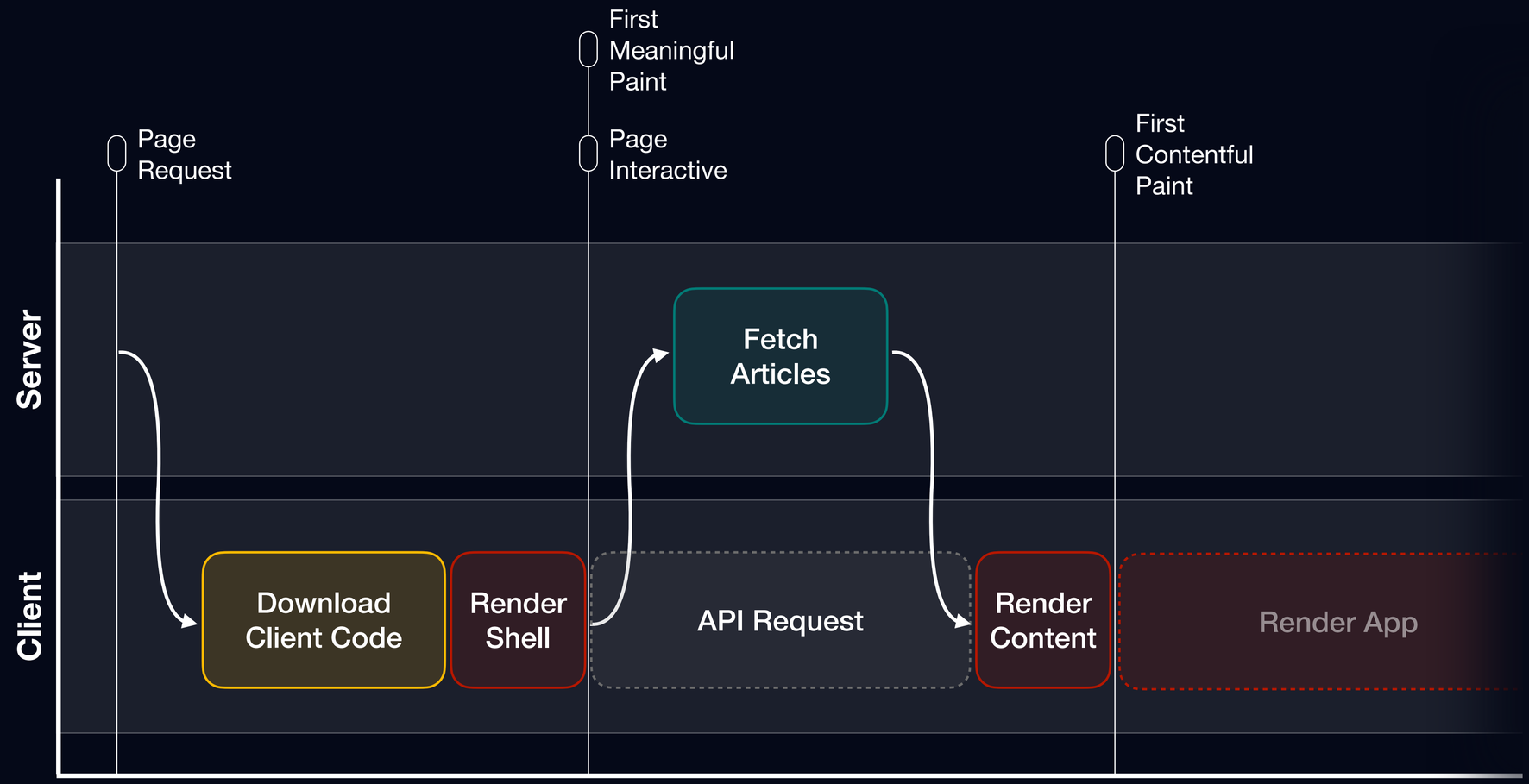
Client Side Rendering



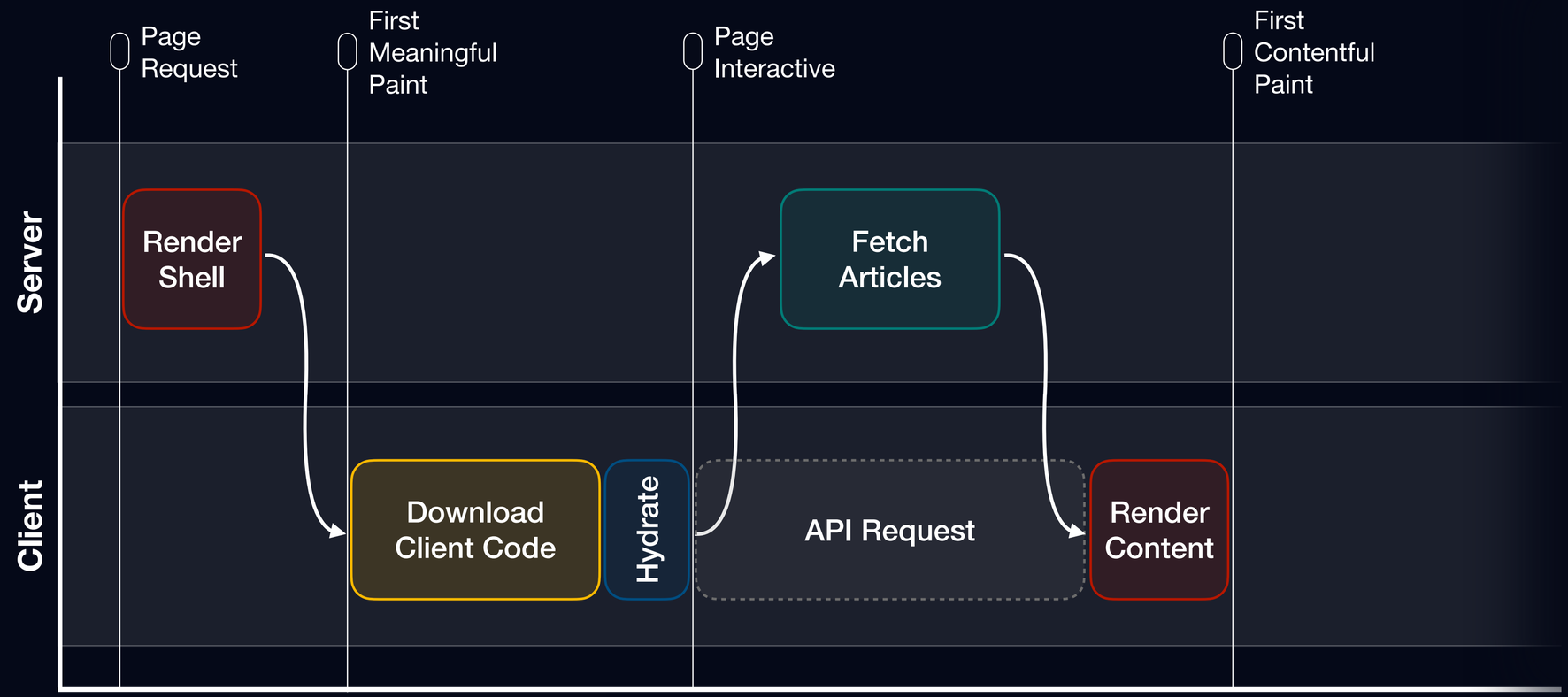
Client Side Rendering



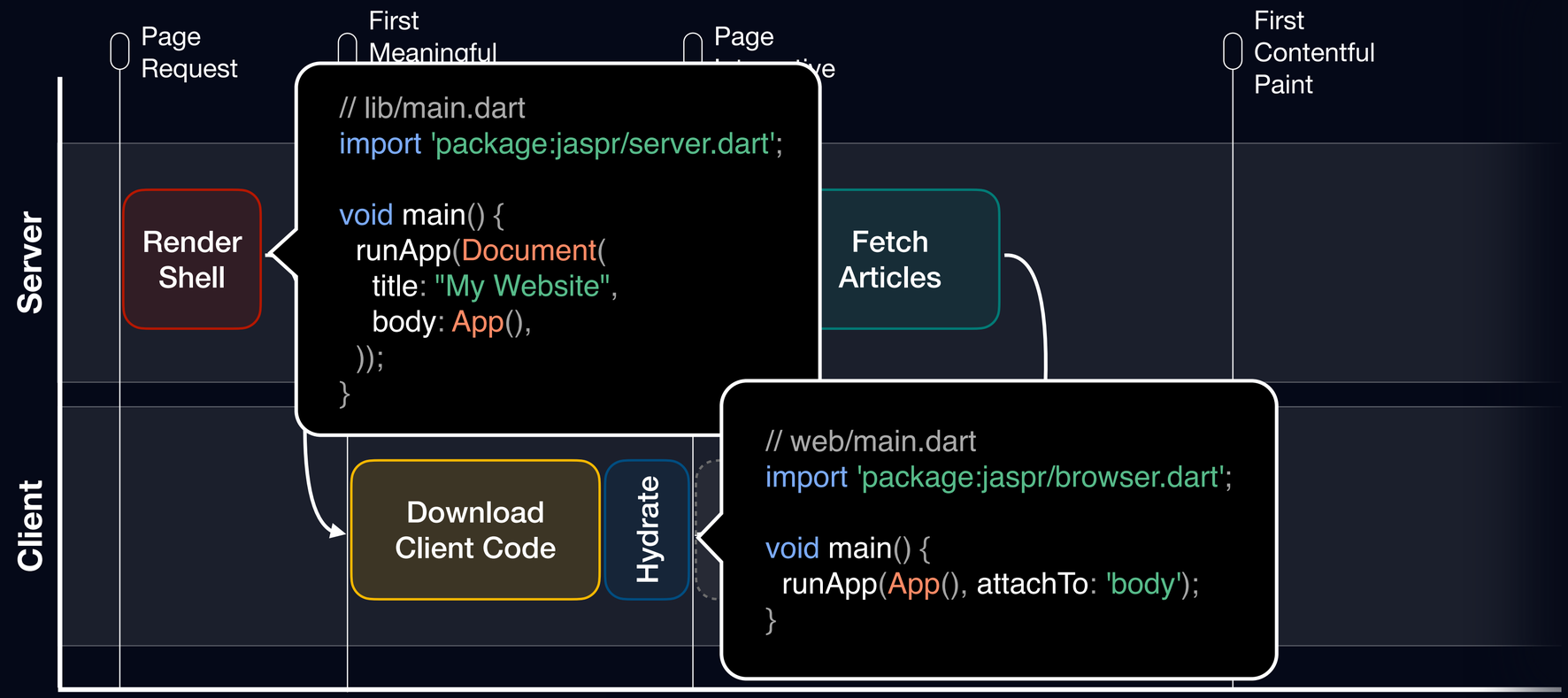
Client Side Rendering



Server Side Rendering



Server Side Rendering



Jasprs Rendering System

Component

FooComponent

Holds config for a piece of UI

Has a public API

Frequently rebuilds

Element

FooElement

Represents an actual piece of the UI

Rarely rebuilds

Client

DOMRenderObject

DOMRenderObject

Controls one plain DOM node

Server

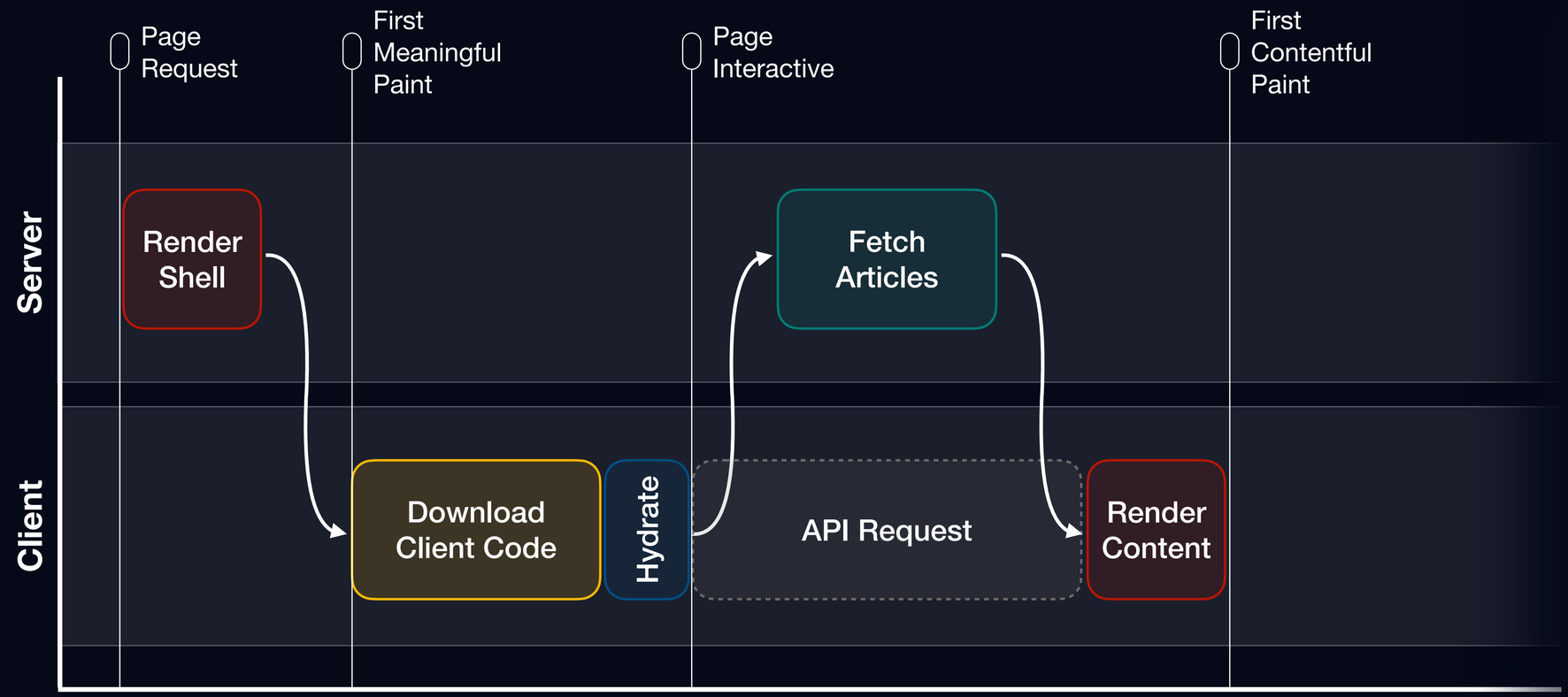
MarkupRenderObject

MarkupRenderObject

Represents a html node
Renders to a String

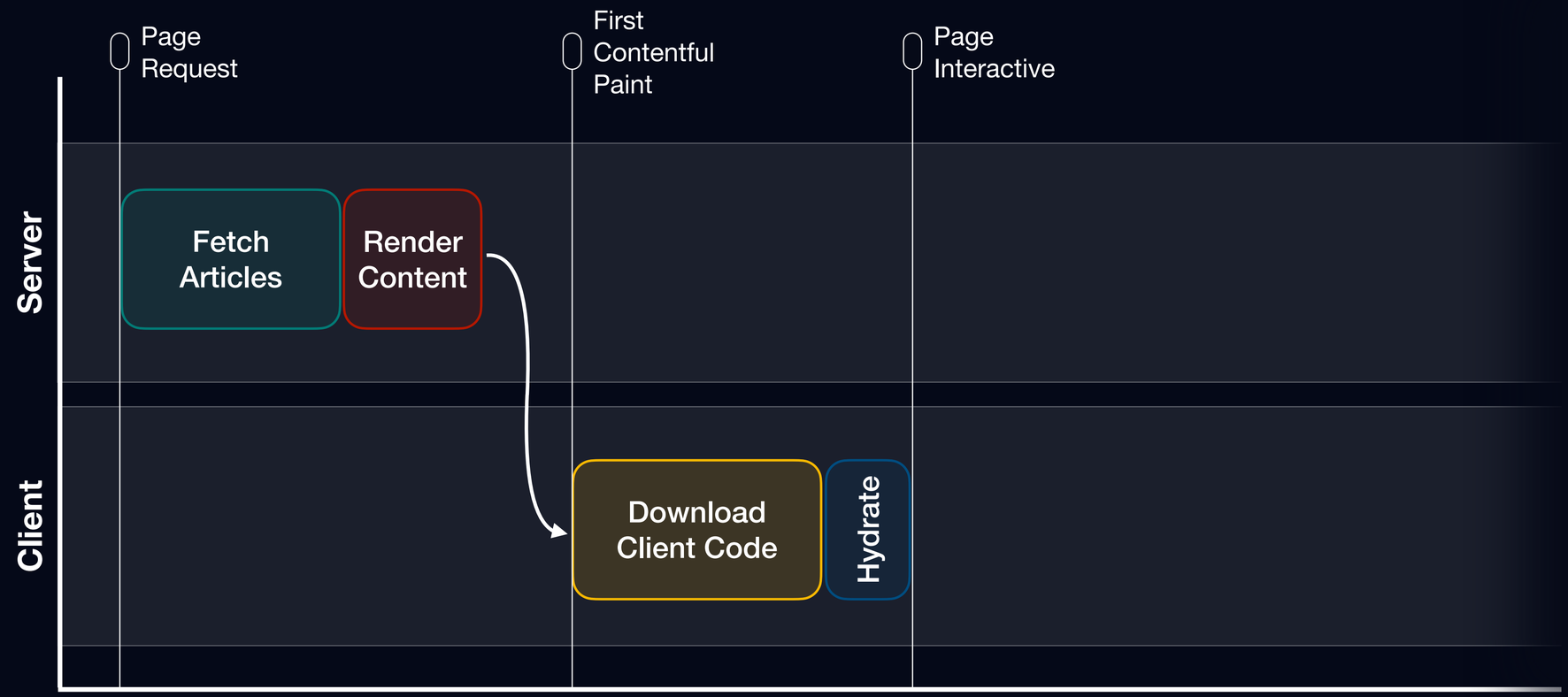
Server Side Rendering

(The dumb way)



Server Side Rendering

(The smart way)



```
class ArticlesList extends AsyncStatelessComponent {
  const ArticlesList({super.key});

  Stream<Component> build(BuildContext context) async* {
    var articles = await fetchArticlesFromDatabase();

    yield ul([
      for (var article in articles)
        li([
          ArticleEntry(article: article),
        ]),
    ]);
  }
}

// AsyncElement

@override
Future<void> performRebuild() async {
  Iterable<Component> _built = await component.build(this).toList();
  /*...*/
}
```

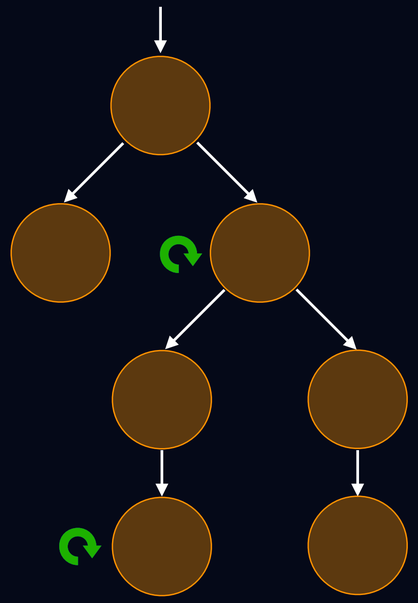
BuildOwner

Controls the rebuilding and calls `element.performRebuild()` on dirty elements.

Server

AsyncBuildOwner

Supports async builds on the server.



Building

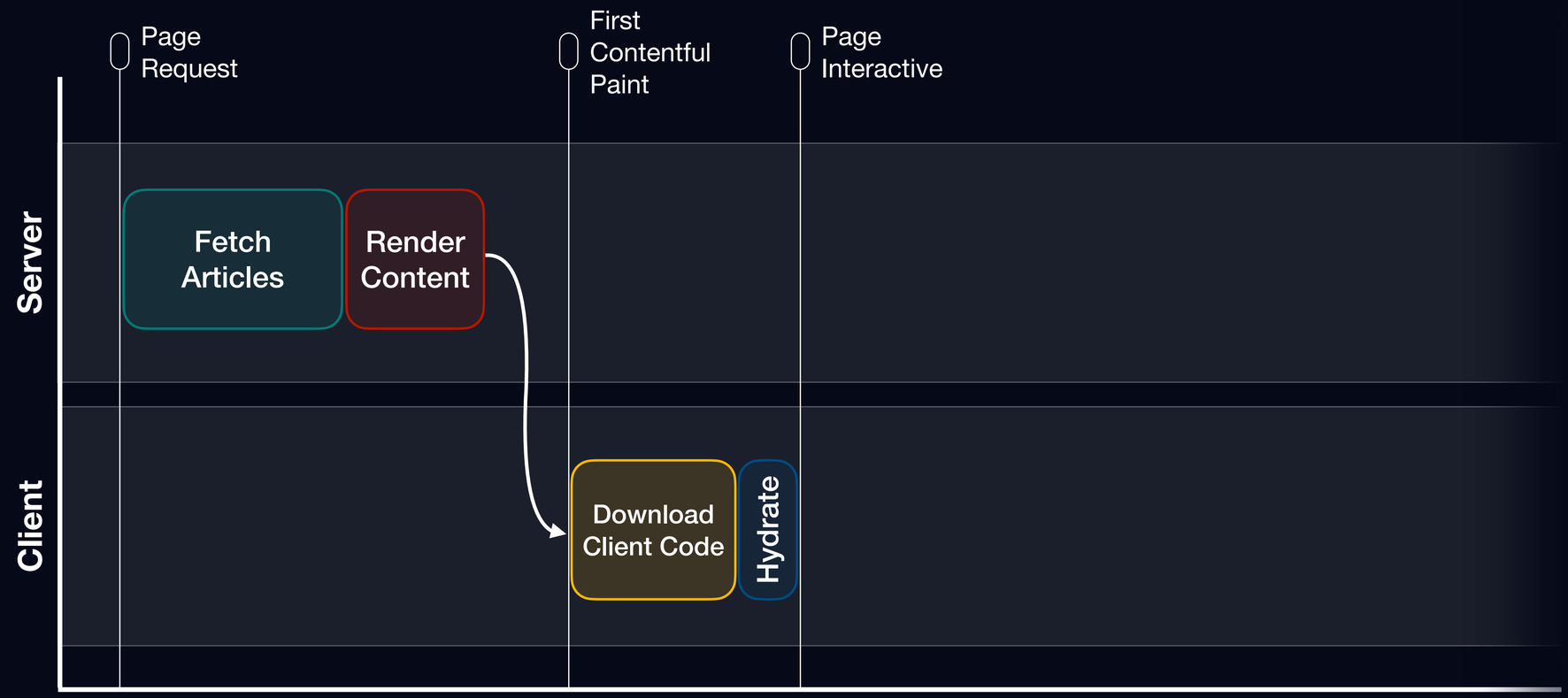
Built

Attached

Pending

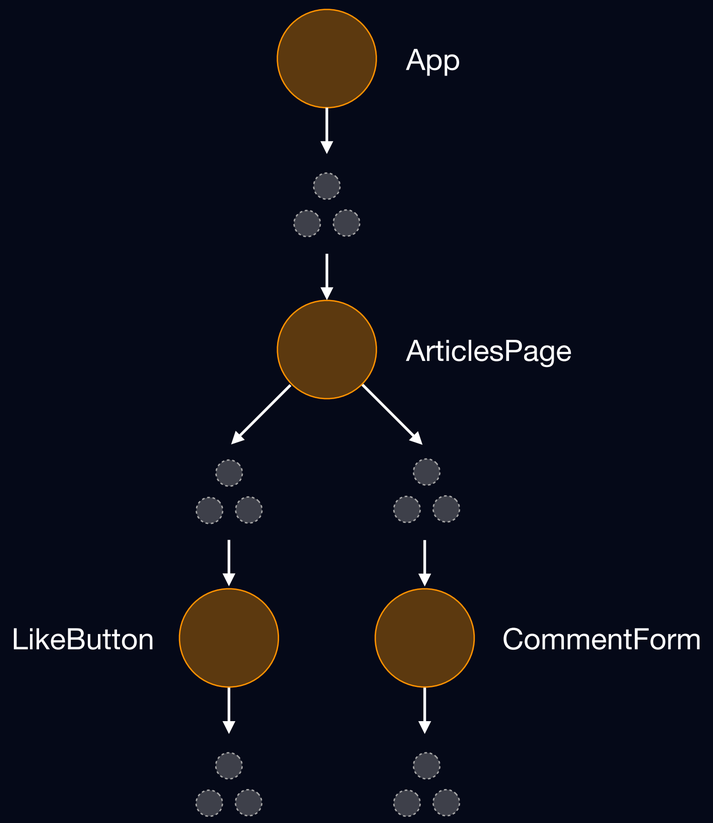
Completed

Server Side Rendering



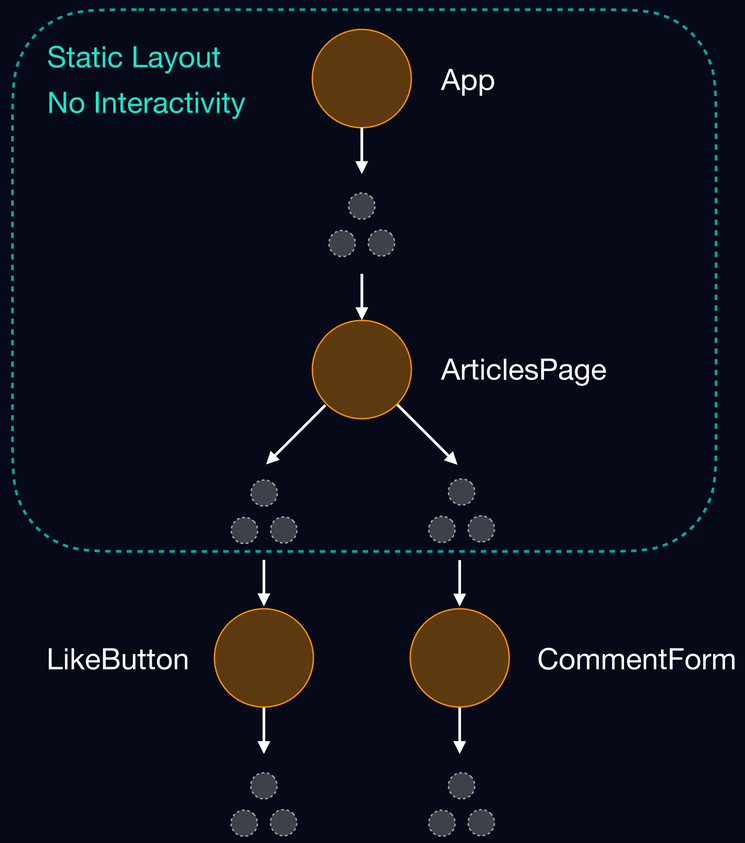
Server

```dart
// lib/main.dart

void main() {
  runApp(Document(
    title: "My Website",
    body: App(),
  ));
}
```



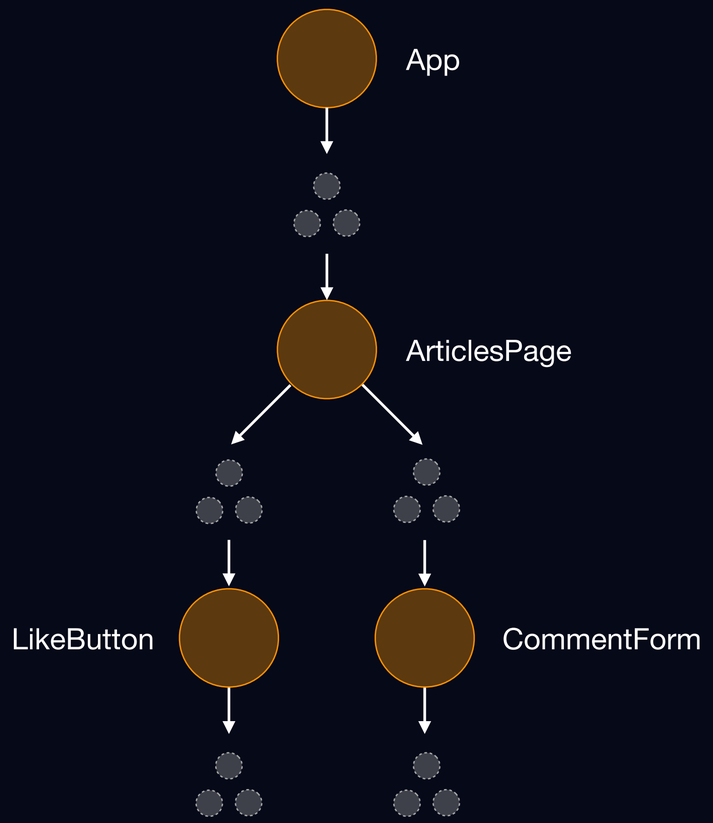
Client

```dart
// web/main.dart

void main() {
  runApp(App(), attachTo: 'body');
}
```



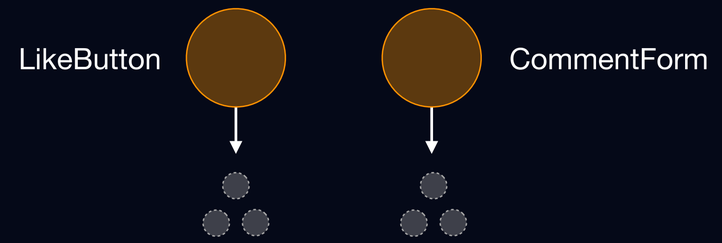
Server

```dart
// lib/main.dart

void main() {
  runApp(Document(
    title: "My Website",
    body: App(),
  ));
}
```



Client

```dart
// lib/components/like_button.dart

@client
class LikeButton extends StatelessComponent {
  const LikeButton({super.key});

  /* ... */
}

// lib/components/comment_form.dart

@client
class CommentForm extends StatelessComponent {
  const CommentForm({super.key});

  /* ... */
}
```



```dart
// lib/components/like_button.dart

@client
class LikeButton extends StatelessComponent {
  const LikeButton({super.key});

  /* ... */
}
```

Generates

```dart
// web/components/like_button.client.dart (generated)

void main() {
  runClient(
    'components/like_button',
    () => LikeButton(),
  );
}
```

Runs

```dart
// package:jaspr

void runClient(String name, Component Function() builder) {
  // Find the „<!—$…-->" marker.
  var marker = findClientMarker(name);

  // Create a new binding and attach the component
  // between the markers.
  BrowserAppBinding().attachRootComponent(
    builder(),
    attachBetween: marker
  )
}
```

Hydrates

```dart
<!--$component/like_button-->
<button>
  Like Me
</button>
<!--/$component/like_button-->
```

Rendering

```dart
// lib/components/like_button.dart

@client
class LikeButton extends StatelessComponent {
  const LikeButton({super.key});

  /* ... */
}
```

Generates

```dart
// web/components/like_button.client.dart (generated)

void main() {
  runClient(
    'components/like_button',
    () => LikeButton(),
  );
}
```

Rendering

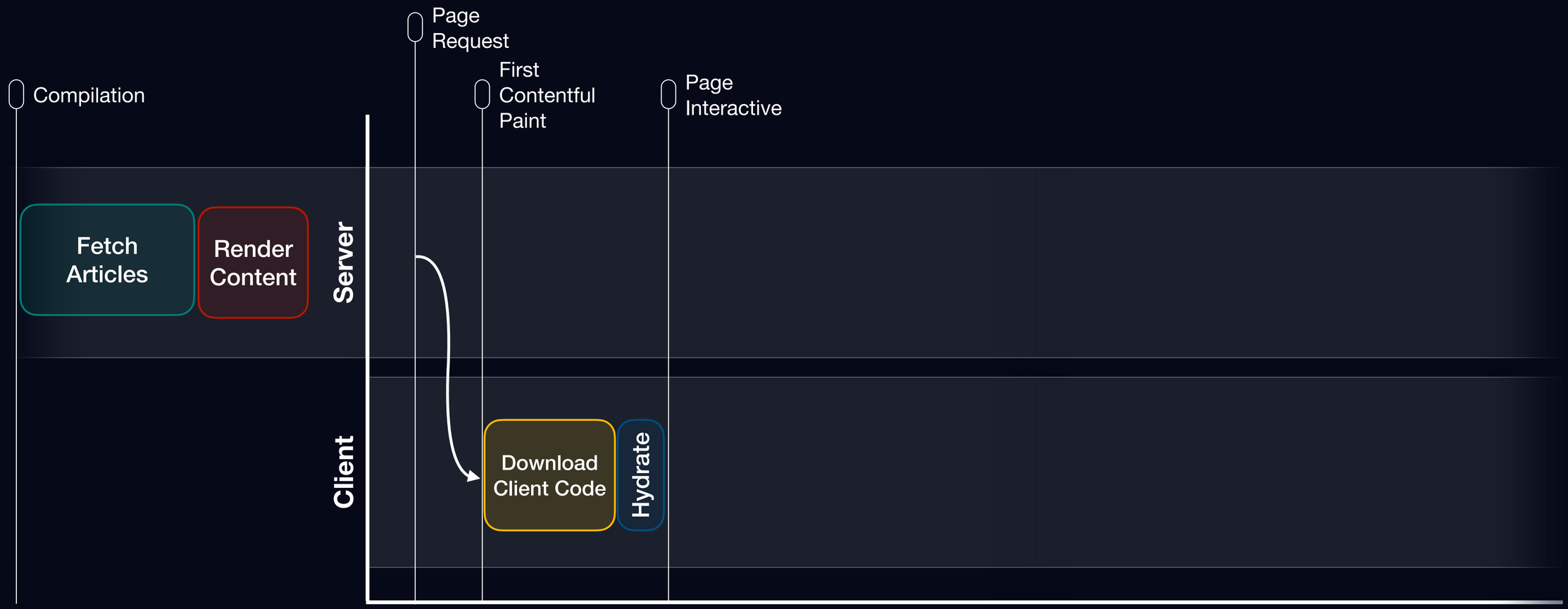
```html
<!--$component/like_button-->
<button>
  Like Me
</button>
<!--/$component/like_button-->
```

Hydrates

Runs

```dart
// package:jaspr

void runClient(String name, Component Function() builder) {
  // Find the „<!—$…-->" marker.
  var marker = findClientMarker(name);

  // Create a new binding and attach the component
  // between the markers.
  BrowserAppBinding().attachRootComponent(
    builder(),
    attachBetween: marker
  )
}
```

Static Site Generation



```
void buildStatic() {

  buildWebDir();
  startServer(release: true);

  /* ... */

  // Generate all routes
  while (queuedRoutes.isNotEmpty) {
    var route = queuedRoutes.removeLast();

    var response = await http.get(Uri.parse('http://localhost:8080$route'));

    File('build/jaspr/$route/index.html')
      ..createSync(recursive: true)
      ..writeAsBytesSync(response.bodyBytes);
  }
}
```

CSS Dart Bindings

SEO

Using Flutter plugins

Routing

CLI

Lots more to discover!

Using custom Backends

Javascript Integrations

Deployment

State Management

Testing

Flutter Embedding

Try it out!

/schultek/jaspr

And tell me!

/schultek_dev