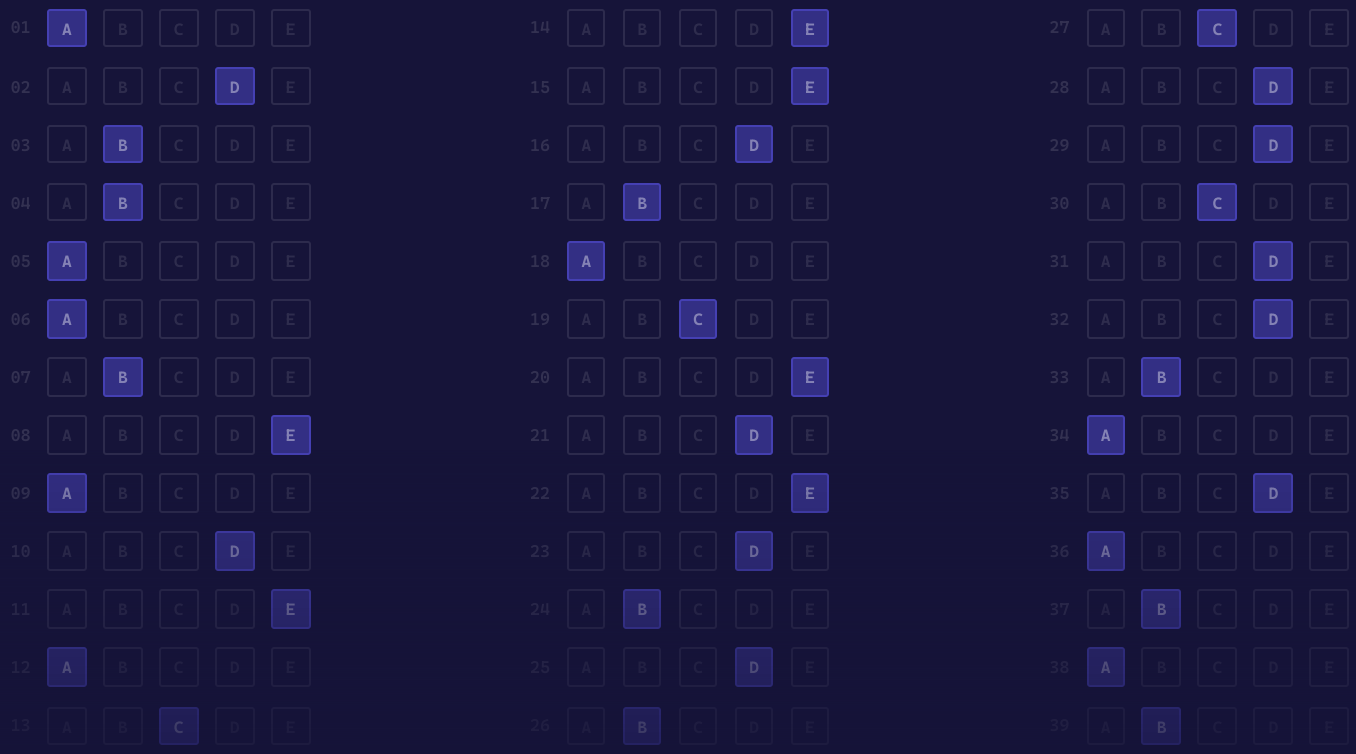
MOMETRIX
SMARTER PREP. BETTER RESULTS.



PRACTICE QUESTIONS

OutSystems

FE11S

Front-end Specialization (ES)

EDITION	QUESTIONS	FORMAT
Demo	9	Limited Edition PDF

WEB www.mometrix.net

SUPPORT support@mometrix.net

Before you begin

What this is

9 practice questions for OutSystems `FE11S`, each with its answer key and an explanation. Answer a question before you read its key — the explanation is worth more once you have committed.

If something looks wrong

Send us three things and we will check it:

- the exam code, `FE11S`
- the question number
- the email address on your order

Write to **support@mometrix.net**. We reply within one business day.

QUESTION 1 SINGLE CHOICE

In an OutSystems Web application, which CSS selector targets an OutSystems-styled button rendered with the class structure used by the default Button widget (e.g., a button named Submit inside a form named MainForm)?

A .MainForm .Submit { ... }

B **.MainForm .btn .btn-default { ... }**

C input[type='button'].MainForm.Submit { ... }

D .Theme1 .MainForm .Submit { ... }

ANSWER **B**

WHY OutSystems wraps Button widgets in nested elements such as .btn and .btn-default inside the screen/form container, so a selector like '.MainForm .btn .btn-default' reliably reaches the rendered button chrome. Option A treats the widget name as a class, which is not how the class chain is generated. Option C combines attribute and class selectors in a way that does not match the generated markup. Option D invents a theme class that is not part of the standard pattern.

QUESTION 2 SINGLE CHOICE

An OutSystems developer wants to run a JavaScript function every time the user clicks a Link widget, without performing a screen navigation. Which approach is the recommended way to do this?

A Add an inline 'onclick' attribute directly on the Link widget in the HTML source.

B **Attach a Client Action with JavaScript code to the Link widget's 'OnClick' event handler.**

C Use a Screen Action with a JavaScript node and call it from the Link's destination.

D Replace the Link widget with an HTML element and bind a custom event in JavaScript.

ANSWER **B**

WHY Client Actions are the supported mechanism for running JavaScript in response to UI events in OutSystems; attaching one to the Link widget's OnClick event executes the script without navigating. Inline 'onclick' attributes are stripped/normalized by the platform and are not the recommended pattern. Screen Actions trigger server roundtrips, and replacing the widget with raw HTML bypasses OutSystems' event wiring model.

QUESTION 3 SINGLE CHOICE

Which OutSystems layout pattern is designed to adapt content to different screen sizes by stacking columns vertically on small viewports and arranging them side-by-side on larger ones?

- **A** **Columns pattern with responsive breakpoints**
- **B** Fixed Grid pattern
- **C** MasterDetail pattern
- **D** Tabs pattern

ANSWER **A**

WHY The Columns pattern is the OutSystems layout element that defines a responsive row of columns which stack on smaller viewports and lay out horizontally on larger ones. Fixed Grid does not reflow responsively in the same way. MasterDetail is for showing parent/child records, not responsive stacking. Tabs is a navigation/content-switching pattern, not a layout grid.

QUESTION 4 SINGLE CHOICE

When configuring an Image widget in OutSystems for accessibility, which property must be set so that screen readers can describe the image to users with visual impairments?

- **A** Tooltip
- **B** **Alternative Text**
- **C** Extended Properties → aria-role
- **D** Style → alt

ANSWER **B**

WHY OutSystems Image widgets expose an 'Alternative Text' property that maps to the underlying HTML 'alt' attribute and is read by assistive technologies. Tooltip is a visual hint, not a reliable accessibility cue. The aria-role extended property controls semantics, not the alternative description. 'Style → alt' is not a valid property path in the widget.

QUESTION 5 SINGLE CHOICE

A developer needs to apply a custom background color to all primary buttons across an entire OutSystems Reactive Web app. Where is the most appropriate place to define this style so it propagates through the theme?

A In the screen's Local Stylesheet for each screen individually.

B **In the Theme's Stylesheet so it cascades to all screens using the theme.**

C In an OnReady Client Action using JavaScript.

D In the eSpace's CSS file imported only by the Home screen.

ANSWER B

WHY Defining the style in the Theme's Stylesheet ensures every screen that uses that theme inherits the change, which is the recommended scoping for cross-screen visual rules. Per-screen Local Stylesheets only affect that single screen. Injecting CSS at runtime via OnReady is fragile and bypasses theme management. Importing CSS on the Home screen does not propagate to other screens.

QUESTION 6 SINGLE CHOICE

An OutSystems form has multiple mandatory Input widgets. Which built-in Input System variable can be checked in the Screen Action's Save logic to determine whether the form is valid before proceeding with the database write?

A FormIsReady

B **Form.Valid**

C Form.IsValid

D FormSubmitted

ANSWER B

WHY In OutSystems Reactive and Traditional Web apps, the Input System exposes a 'Form.Valid' boolean that aggregates the validation state of mandatory and typed inputs; it returns true only when all constraints are satisfied. 'Form.IsValid' is not a documented member of the Form variable. 'FormIsReady' and 'FormSubmitted' are not part of the Input System's API.

QUESTION 7 SINGLE CHOICE

A developer binds a List widget to a local variable that holds a list of records. When the user clicks an item, the selected row should be highlighted and a detail panel populated. Which OutSystems approach correctly captures the clicked item inside the List Item's OnClick handler?

A **Use the built-in 'ListItem.Current' inside the List Item's expression/event context.**

B Use a global variable assigned from the screen OnReady.

C Read the DOM index using JavaScript and compute the item manually.

D Use the screen's 'List.Current' variable outside the List widget.

ANSWER **A**

WHY Inside a List Item's scope, OutSystems automatically exposes 'ListItem.Current' (and equivalents) which represents the row being interacted with, and it is the supported way to reference the clicked item. Global variables do not receive the row context automatically. Reaching into the DOM to compute indices is fragile and unsupported. 'List.Current' is not exposed outside the list item scope and cannot be used to infer the clicked row.

QUESTION 8 SINGLE CHOICE

In an OutSystems Reactive Web application, which client-side action node is used to programmatically navigate the user to another screen while preserving the browser history?

A Navigate (built-in client action)

B **Submit and Navigate (built-in client action)**

C Redirect (built-in client action)

D Window.Open (JavaScript node)

ANSWER **B**

WHY 'Submit' (or 'Submit and Navigate' in some versions) is the client action that performs a logical navigation to a destination screen, typically via the platform's client routing, which preserves the standard SPA history model. The plain 'Navigate' node alone is more associated with screen action flows and server-side navigation. 'Redirect' issues a hard browser redirect that replaces history. Window.Open opens a new browser window/tab, not in-app navigation.

QUESTION 9 MULTIPLE CHOICE

A developer needs to customize the visual appearance of an OutSystems Reactive Web application so that it follows the company's brand guidelines. Which of the following are valid approaches to apply custom CSS in an OutSystems Reactive Web app? (Choose two.)

A **Add a custom CSS class to a specific widget using the 'Style Classes' property and define the styles in the app's theme.**

B Embed an inline <style> block inside a screen's HTML content placeholder to override theme rules.

C **Import a third-party CSS framework by referencing it through the 'Resources' section and adding the link to the screen's head tag.**

D Modify the generated JavaScript bundle directly in Service Studio's compiled output folder to inject styles.

ANSWER **A · C**

WHY In OutSystems Reactive Web apps, the recommended way to apply custom styling is to define classes in the app's theme and reference them via the 'Style Classes' property on a widget (option A). Importing external CSS files is also supported through the Resources module and adding the link in the screen's Head placeholder (option C). Inline <style> blocks inside HTML content are not a recommended OutSystems pattern and Service Studio does not expose a way to modify the compiled JavaScript bundle directly, so B and D are incorrect.