Demonstrate ability to use layout and XML schema

Describe the elements of the Magento layout XML schema, including the major XML directives.

Xsi:type niceties

string:

```
<argument xsi:type="string" translate="bool">
</argument>
```

- true values: [true, 1, 'true', '1']
- false values: [false, 0, 'false', '0']

options:

```
<argument name="options" xsi:type="options"
model="Magento\Tax\Model\TaxClass\Source\Customer"/>
```

- Model class instance of Data\OptionSourceInterface - toOptionArray.
- Result = model.toOptionArray, slightly processed, optgroup will probably not work.

url:

```
<argument xsi:type="url" path="">
    <param name="_current">1</param>
</argument>
```

```
<argument name="configuration" xsi:type="array">
    <item name="recently_viewed_product" xsi:type="array">
        <item name="requestConfig" xsi:type="array">
            <item name="syncUrl" xsi:type="url"
path="catalog/product/frontend_action_synchronize"/>
        </item>
    </item>
</argument>
```

helper:

```
<action method="setTemplate">
    <argument name="template" xsi:type="helper"
helper="Magento\Paypal\Helper\Hss::getReviewButtonTemplate">
        <param name="name">Magento_Paypal::hss/review/button.phtml</param>
    </argument>
</action>
```

Major directives:

Node readers:

- `<html>`
- `<head>`
- `<body>`
- `<move>`
- `<container>`
- `<block>`
- `<uiComponent>`

Generators:

- `<head>`
- `<body>`

- `<container>`
- `<block>`
- `<uiComponent>`

Scheduled Structure - View\Page\Config\Structure:

- `scheduledPaths[$blockName] = $parent1/$parent2/$blockName`
- `scheduledStructure[$blockName] = $row`
- `scheduledData[$blockName] = [attributes actions arguments]`
- `elementAttributes[element][]`
- `removeElementAttributes[element][attribute]`
- `assets[/original/link/src] = attributes`
- `removeAssets[/some/src/] = /some/src`
- `title = __('Value')`
- `metadata[name] = content`
- `bodyClasses[]`

`<HTML>` node reader View\Page\Config\Reader\Html

Html reader is used for collecting attributes of html in to the scheduled page structure.

```
<html>
    <attribute name="html_attribute_name" value="html_attribute_value"
/>
</html>
```

structure:

`elementAttributes['html']['html_attribute_name'] = 'html_attribute_value'`

render:

- View\Result\Page::render
- `'htmlAttributes' => $this->pageConfigRenderer->renderElementAttributes($config::ELEMENT_TYPE_HTML)`
- View\Page\Config\Renderer::renderElementAttributes - join(' ', name-value pairs)
- root.phtml

  ```
  <!doctype html>
  <html <?= /* @escapeNotVerified */ $htmlAttributes ?
  >>
  ```

`<HEAD>` node reader View\Page\Config\Reader\Head

- `<css src="" ...attributes>`, `<script src="" ...attributes>`, `<link src="" ...attributes>` View\Page\Config\Structure::addAssets: structure `assets['path/to/src'] = attributes`

- `<remove src="">` structure.removeAssets: structure `removeAssets['path/to/src'] = 'path/to/src'`

- `<title>` structure.setTitle `title = __($title)`

- `<meta name="" content="">` or `<meta property="" content="">` structure.setMetadata: `metadata[name] = content`

- `<attribute name value />` structure.setElementAttribute `elementAttributes[head][attribute]=value`

  render:

  - View\Result\Page::render

    - `'headAttributes' => $this->pageConfigRenderer->renderElementAttributes($config::ELEMENT_TYPE_`
    - root.phtml: `<head <?= /* @escapeNotVerified */ $headAttributes ?>>` - key1=value1 key2=value2

`<HEAD>` generator View\Page\Config\Generator\Head

- processRemoveAssets - structure. `assets` vs structure. `removeAssets`
- processRemoveElementAttributes - `elementAttributes` vs `removeElementAttributes`
- processAssets - `assets`
    - add *remote* assets, attribute src_type='controller' or src_type='url'
        - example `<css src="" src_type="controller" content_type="link" />`
        - Page\Config::addRemotePageAsset
    - add normal assets View\Page\Config::addPageAsset
- processTitle - copy from structure to page config
- processMetadata - copy
- processElementAttributes - copy: html, head etc.

`<BODY>` node reader View\Page\Config\Reader\Body

```
<body>
    <attribute name="id" value="html-body"/>
    <attribute name="class" value="preview-
window"/>
</body>
```

- "class" attributes - structure. `bodyClasses[] = class-name` tip: empty class clears all
- other attributes - structure. `elementAttributes[body][attribute-name] = value`
    - View\Result\Render `'bodyAttributes' => $this->pageConfigRenderer->renderElementAttributes($config::ELEMENT_TYPE_`
    - root.phtml `<body ... <?= /* @escapeNotVerified */ $bodyAttributes ?>>`
- calls readerPool to process `<body>` contents recursively

`<BODY>` generator View\Page\Config\Generator\Body

Copies structure. `bodyClasses[]` to View\Page\Config::addBodyClass

- adds to pageConfig. `elements[body][class] = '...'`

`<MOVE>` node reader View\Layout\Reader\Move

```
<move element='' destination='' as='' (after=''|before='') />
```

structure. `scheduledMoves[elementName] = [destination, siblingName, isAfter, alias]`

All moves are processed in View\Layout\GeneratorPool before all other generators:

- generatorPool.buildStructure
    - scheduleElement - creates array structure `_elements`
        - `_elements[parentId]['children'][childId] = alias`
        - `_elements[childId]['parent'][parentId] = alias`
        - bad relations `brokenParent[elementId] = 1`
    - reorderElements
    - moveElementInStructure – here
    - removeElement - broken parents
    - remove elements when 'visibilityConditions' doesn't match

`<CONTAINER>`, `<REFERENCECONTAINER>` node reader View\Layout\Reader\Container

- `<container>`:

    - `<container name as before/after>` - schedule structure structure.
      `scheduledStructure[name] = [container alias parent sibling is_after]`
    - containerReader.mergeContainerAttributes structure.
      `scheduledData[name][attributes] = [tag id class label display]`
- `<referenceContainer name remove ...attributes>`
    - possible to un-remove
    - merges normal container attributes
- calls readerPool to process contents recursively, just like `<body>`

`<BLOCK>` node reader View\Layout\Reader\Block

- schedules structure View\Layout\ScheduledStructure\Helper::scheduleStructure
    - generates name if needed - like `{$parent_name}_schedule_block1`
    - [type='block', alias='$as', parent_name='$parent', sibling_name='$beforeAfterName', is_afer=true/false]
    - `scheduledPaths[$blockName] = $parent1/$parent2/$blockName`
    - `scheduledStructure[$blockName] = $row`
- data['attributes'] - [class, group, template, ttl, display, acl]
- data['actions'][] = [method, arguments, ifconfig, scope=STORE]
- data['arguments'] = parse `<arguments>` :
    - `<argument name="$name">...</argument>` View\Layout\Argument\Parser::parse Config\Converter\Dom\Flat::convert

```
<arguments>
    <argument>Text node</argument>
    <argument>
        <item>
          <item>...parsed
recursively</item>
        </item>
    </argument>
</arguments>
```

- blockReader.evaluateArguments
    - Data\Argument\InterpreterInterface::evaluate
    - Data\Argument\Interpreter\Composite as layoutArgumentReaderInterpreter

```
<argument name="interpreters" xsi:type="array">
    <item name="options"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\Options</item>
    <item name="array"
xsi:type="object">layoutArrayArgumentReaderInterpreterProxy</item>
    <item name="boolean"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\Boolean</item>
    <item name="number"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\Number</item>
    <item name="string"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\StringUtils</item>
    <item name="null"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\NullType</item>
    <item name="object"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\Passthrough</item>
    <item name="url"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\Passthrough</item>
    <item name="helper"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\Passthrough</item>
</argument>
```

Scheduled Structure:

- `scheduledPaths[$blockName] = $parent1/$parent2/$blockName`
- `scheduledStructure[$blockName] = $row`
- `scheduledData[$blockName] = [attributes actions arguments]`

`<BLOCK>` generator View\Layout\Generator\Block

View\Layout\Generator\Block::process: - creates block instance, evaluates arguments

- finds scheduled elements with type = ‘block’
- generator.generateBlock
    - Data\Argument\InterpreterInterface::evaluate

  View\Layout\Argument\Interpreter\Decorator\Updater as layoutArgumentGeneratorInterpreter
    - modifies result via registered View\Layout\Argument\UpdaterInterface

```
<argument name="testArrayWithUpdater" xsi:type="array">
    <updater>Magento\Framework\View\Layout\Argument\UpdaterInterface</updater>
    <item name="add" xsi:type="array">
        <item name="label" xsi:type="string" translate="true">Move to
Archive</item>
        <item name="url" xsi:type="string">*/sales_archive/massAdd</item>
    </item>
</argument>
<argument name="dataSource" xsi:type="object">
    <updater>Magento\Sales\Model\Grid\CollectionUpdater</updater>
</argument>
```

Data\Argument\Interpreter\Composite as layoutArgumentGeneratorInterpreterInternal

```
<argument name="interpreters" xsi:type="array">
    <item name="options"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\Options</item>
    <item name="array"
xsi:type="object">layoutArrayArgumentGeneratorInterpreterProxy</item>
    <item name="boolean"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\Boolean</item>
    <item name="number"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\Number</item>
    <item name="string"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\StringUtils</item>
    <item name="null"
xsi:type="object">Magento\Framework\Data\Argument\Interpreter\NullType</item>
    <item name="object" xsi:type="object">layoutObjectArgumentInterpreter</item>
    <item name="url"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\Url</item>
    <item name="helper"
xsi:type="object">Magento\Framework\View\Layout\Argument\Interpreter\HelperMethod</item>
</argument>
```

- every block:
    - block.setLayout -> _prepareLayout
    - event `core_layout_block_create_after`
- every scheduled action:
    - generator.generateAction - evaluate arguments and call_user_func_array

`<UI_COMPONENT>` node reader View\Layout\Reader\UiComponent

```
<ui_component name group aclResource ifconfig>
    <visibilityCondition name="" className="">
        <arguments>...</arguments>
    </visibilityCondition>
</ui_component>
```

- scheduleStructure - structure.
  `scheduledStructure[name] = [uiComponent alias parent sibling is_after]`
- attributes: group, component, aclResource + visibilityConditions structure.
  `scheduledData[name] = [attributes]`
- read ui config, merge definition
    - *magic here*
- ui component generated layout elements -> - calls readerPool to process contents recursively

`<UI_COMPONENT>` generator View\Layout\Generator\UiComponent

- finds all scheduled elements of type uiComponent
- generateComponent - creates components, prepares recursively, wraps in layout block
  - View\Element\UiComponentFactory::create - View\Element\UiComponentInterface compatible with BlockInterface
  - prepare component and its children recursively - UiComponent.prepare - providers, buttons, form data etc.
  - creates layout block wrapper for component - \Magento\Ui\Component\Wrapper\UiComponent - just renders component

How do you use layout XML directives in your customizations?

- move elements
- referenceContainer - change attributes, add children
- referenceBlock - change template, position before/after, add arguments, add children, set no display
- add HEAD scripts/styles

Describe how to create a new layout XML file.

- Place layout in one of directories:
  - custom-module/view/base/layout/
  - custom-module/view/frontend/layout/
  - custom-theme/Magento_Checkout/layout/

  Or the same with page_layout directory.

- When extending page layout, don't copy-paste layout="…" attribute if not intended.

- Use standard XML namespaces:

```
<layout xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:noNamespaceSchemaLocation="urn:magento:framework:View/Layout/etc/layout_generic.xsd">
```

```
<page
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:noNamespaceSchemaLocation="urn:magento:framework:View/Layout/etc/page_configuration.xsd"
>
```

Describe how to pass variables from layout to block.

- use block arguments:

```
<block class="Magento\Customer\Block\Account\Navigation" name="top.links">
    <arguments>
        <argument name="css_class" xsi:type="string">header
links</argument>
    </arguments>
</block>
```

Magento docs

- substitute dynamic argument value as result of helper execution

```
<action method="setTemplate">
    <argument name="template" xsi:type="helper"
helper="Magento\Paypal\Helper\Hss::getReviewButtonTemplate">
        <param name="name">Magento_Paypal::hss/review/button.phtml</param>
    </argument>
</action>
```