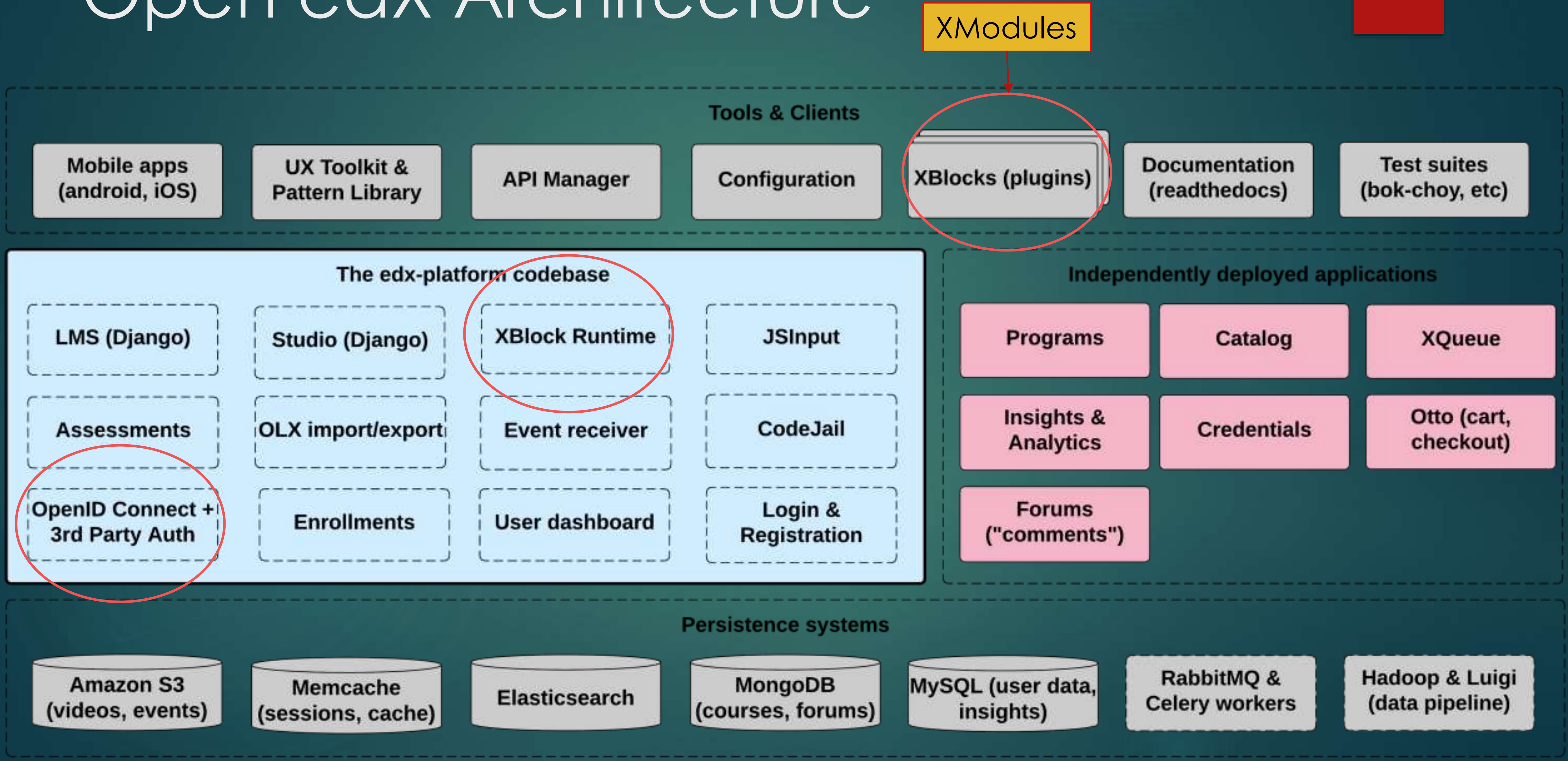
XBlock dev

- rshen@dronsoft.com

Open edX Architecture



XModules

Tools & Clients

Mobile apps (android, iOS)

UX Toolkit & Pattern Library

API Manager

Configuration

XBlocks (plugins)

Documentation (readthedocs)

Test suites (bok-choy, etc)

The edx-platform codebase

LMS (Django)

Studio (Django)

XBlock Runtime

JSInput

Assessments

OLX import/export

Event receiver

CodeJail

OpenID Connect + 3rd Party Auth

Enrollments

User dashboard

Login & Registration

Independently deployed applications

Programs

Catalog

XQueue

Insights & Analytics

Credentials

Otto (cart, checkout)

Forums ("comments")

Persistence systems

Amazon S3 (videos, events)

Memcache (sessions, cache)

Elasticsearch

MongoDB (courses, forums)

MySQL (user data, insights)

RabbitMQ & Celery workers

Hadoop & Luigi (data pipeline)

XBlock

https://edx.readthedocs.io/projects/xblock-tutorial/en/latest/

- https://github.com/edx/XBlock
  - 1.2.9 2019-11-14
  - 1.2.2 2018-8-29
- **https://github.com/edx/xblock-sdk**
  - v0.1.8 2020-3-13
    - python3, django 2.2.10, xblock 1.2.9
  - **v0.1.7 2019-11-28**
    - python2, django 1.11.26, xblock 1.2.9
- bitnami edx 2.4
  - python2, django 1.11.21, xblock 1.2.2

ViewCounter

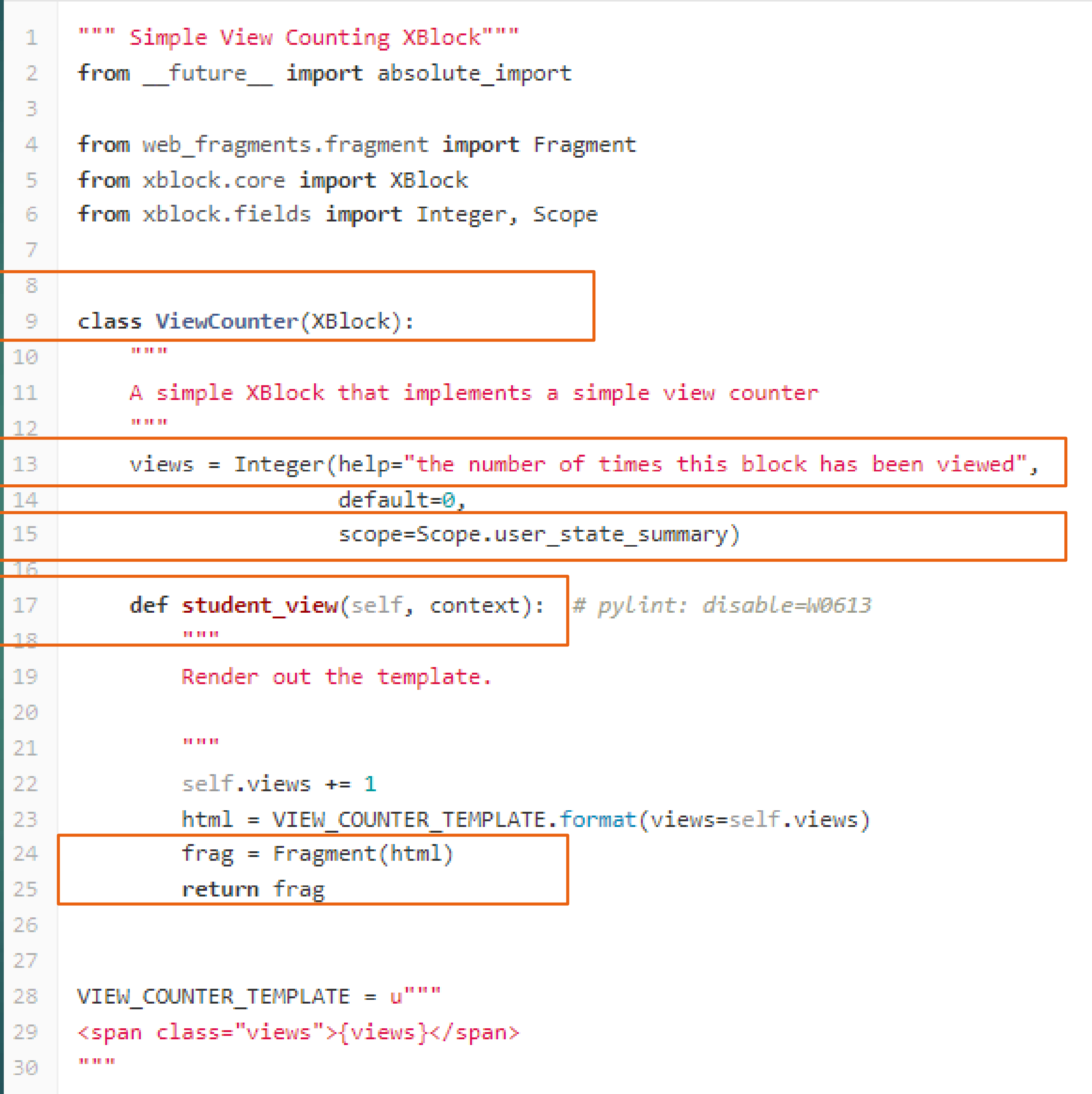
xblock.core.XBlock

xblock.fields.Integer

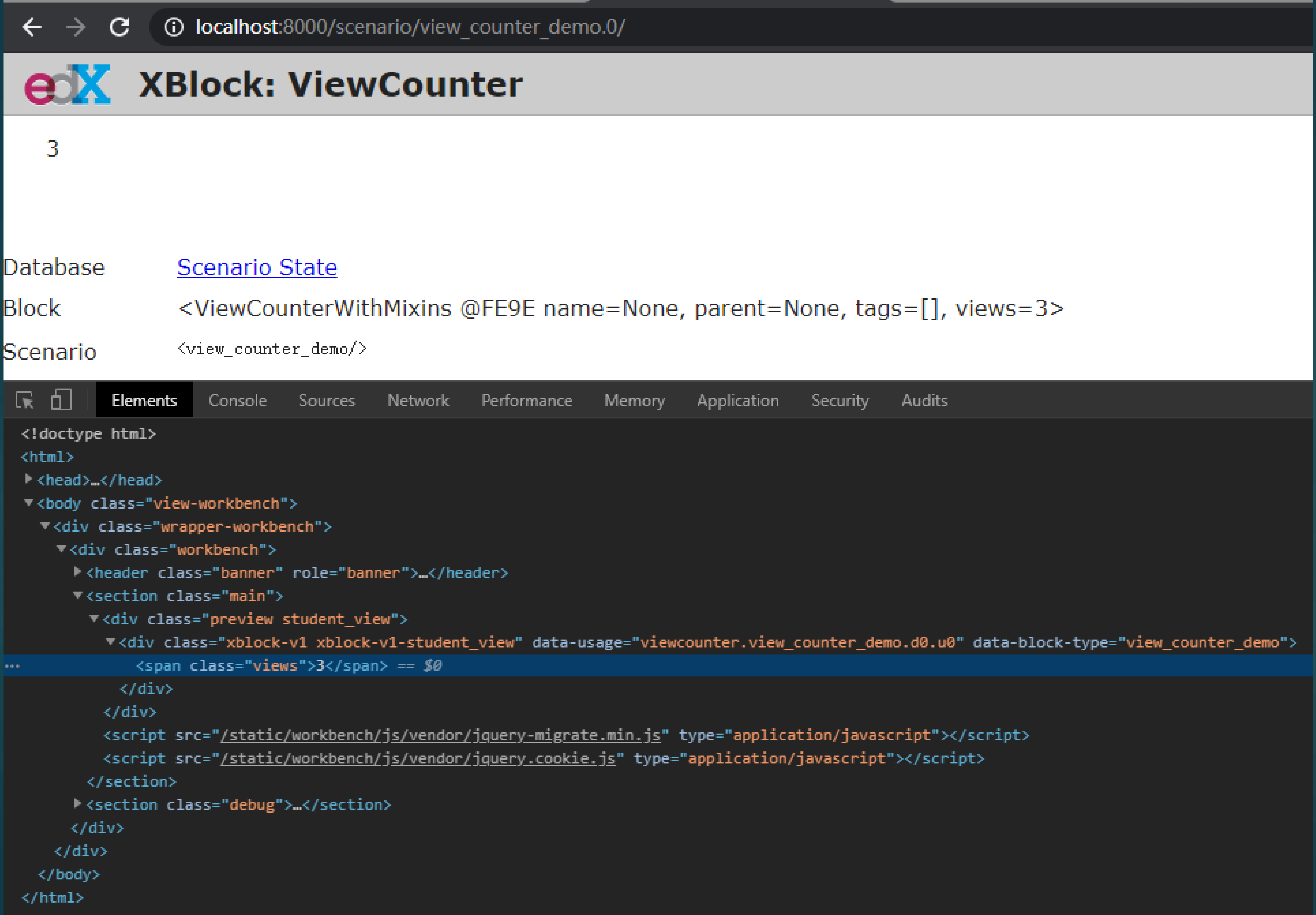
xblock.fields.Scope

student_view

web_fragments.fragment.Fragment
- html
- css
- javascript



```python
""" Simple View Counting XBlock"""
from __future__ import absolute_import

from web_fragments.fragment import Fragment
from xblock.core import XBlock
from xblock.fields import Integer, Scope


class ViewCounter(XBlock):
    """
    A simple XBlock that implements a simple view counter
    """
    views = Integer(help="the number of times this block has been viewed",
                    default=0,
                    scope=Scope.user_state_summary)

    def student_view(self, context):  # pylint: disable=W0613
        """
        Render out the template.

        """
        self.views += 1
        html = VIEW_COUNTER_TEMPLATE.format(views=self.views)
        frag = Fragment(html)
        return frag


VIEW_COUNTER_TEMPLATE = u"""
<span class="views">{views}</span>
"""
```



XBlock: ViewCounter

3

Database Scenario State

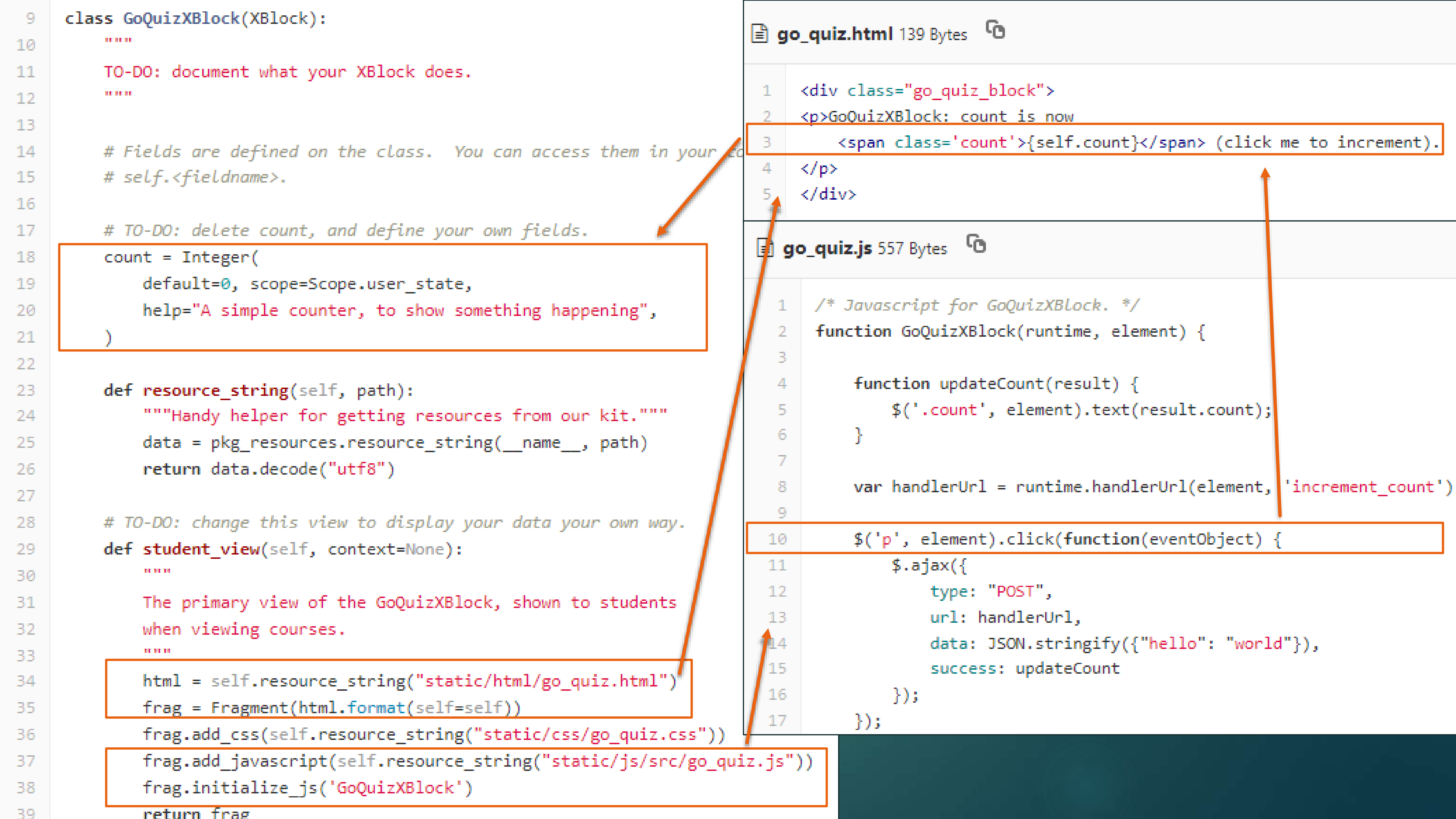
Block <ViewCounterWithMixins @FE9E name=None, parent=None, tags=[], views=3>

Scenario `<view_counter_demo/>`

```
<!doctype html>
<html>
 ▶<head>…</head>
 ▼<body class="view-workbench">
   ▼<div class="wrapper-workbench">
     ▼<div class="workbench">
       ▶<header class="banner" role="banner">…</header>
       ▼<section class="main">
         ▼<div class="preview student_view">
           ▼<div class="xblock-v1 xblock-v1-student_view" data-usage="viewcounter.view_counter_demo.d0.u0" data-block-type="view_counter_demo">
...           <span class="views">3</span> == $0
            </div>
          </div>
          <script src="/static/workbench/js/vendor/jquery-migrate.min.js" type="application/javascript"></script>
          <script src="/static/workbench/js/vendor/jquery.cookie.js" type="application/javascript"></script>
        </section>
       ▶<section class="debug">…</section>
      </div>
    </div>
  </body>
</html>
```

xblock-sdk (Ubuntu 16)

```
$ sudo apt-get install git python-virtualenv
$ mkdir xblock-dev && cd xblock-dev
$ virtualenv venv
$ source venv/bin/activate
(venv) $ git clone https://github.com/edx/xblock-sdk.git -b v0.1.7
(venv) $ cd xblock-sdk
(venv) $ pip install –r requirements/base.txt .
    # setuptools=42.0.1
(venv) $ ./manage.py migrate
(venv) $ cd ..
(venv) $ xblock-sdk/bin/workbench-make-xblock
    Short name: go_quiz             (package name)
    Class name: GoQuizXBlock        (class name)
(venv) $ pip install -e go_quiz
(venv) $ cd xblock-sdk && ./manage.py runserver
```



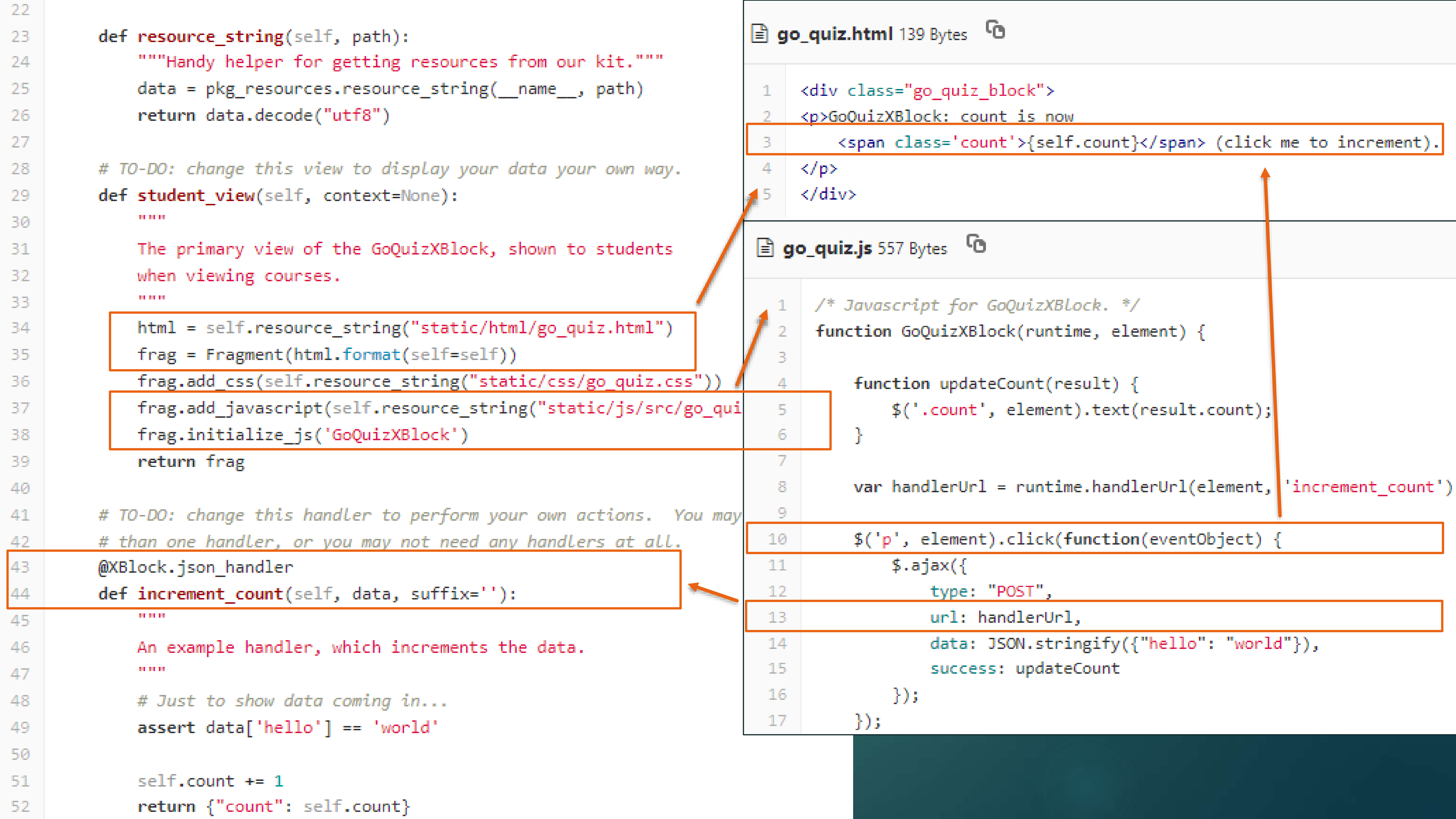
```python
class GoQuizXBlock(XBlock):
    """
    TO-DO: document what your XBlock does.
    """

    # Fields are defined on the class.  You can access them in your co
    # self.<fieldname>.

    # TO-DO: delete count, and define your own fields.
    count = Integer(
        default=0, scope=Scope.user_state,
        help="A simple counter, to show something happening",
    )

    def resource_string(self, path):
        """Handy helper for getting resources from our kit."""
        data = pkg_resources.resource_string(__name__, path)
        return data.decode("utf8")

    # TO-DO: change this view to display your data your own way.
    def student_view(self, context=None):
        """
        The primary view of the GoQuizXBlock, shown to students
        when viewing courses.
        """
        html = self.resource_string("static/html/go_quiz.html")
        frag = Fragment(html.format(self=self))
        frag.add_css(self.resource_string("static/css/go_quiz.css"))
        frag.add_javascript(self.resource_string("static/js/src/go_quiz.js"))
        frag.initialize_js('GoQuizXBlock')
        return frag
```

go_quiz.html 139 Bytes

```html
<div class="go_quiz_block">
<p>GoQuizXBlock: count is now
    <span class='count'>{self.count}</span> (click me to increment).
</p>
</div>
```

go_quiz.js 557 Bytes

```javascript
/* Javascript for GoQuizXBlock. */
function GoQuizXBlock(runtime, element) {

    function updateCount(result) {
        $('.count', element).text(result.count);
    }

    var handlerUrl = runtime.handlerUrl(element, 'increment_count')

    $('p', element).click(function(eventObject) {
        $.ajax({
            type: "POST",
            url: handlerUrl,
            data: JSON.stringify({"hello": "world"}),
            success: updateCount
        });
    });
```



```python

    def resource_string(self, path):
        """Handy helper for getting resources from our kit."""
        data = pkg_resources.resource_string(__name__, path)
        return data.decode("utf8")

    # TO-DO: change this view to display your data your own way.
    def student_view(self, context=None):
        """
        The primary view of the GoQuizXBlock, shown to students
        when viewing courses.
        """
        html = self.resource_string("static/html/go_quiz.html")
        frag = Fragment(html.format(self=self))
        frag.add_css(self.resource_string("static/css/go_quiz.css"))
        frag.add_javascript(self.resource_string("static/js/src/go_qui
        frag.initialize_js('GoQuizXBlock')
        return frag

    # TO-DO: change this handler to perform your own actions.  You may
    # than one handler, or you may not need any handlers at all.
    @XBlock.json_handler
    def increment_count(self, data, suffix=''):
        """
        An example handler, which increments the data.
        """
        # Just to show data coming in...
        assert data['hello'] == 'world'

        self.count += 1
        return {"count": self.count}
```

go_quiz.html 139 Bytes

```html
<div class="go_quiz_block">
<p>GoQuizXBlock: count is now
    <span class='count'>{self.count}</span> (click me to increment).
</p>
</div>
```

go_quiz.js 557 Bytes

```javascript
/* Javascript for GoQuizXBlock. */
function GoQuizXBlock(runtime, element) {

    function updateCount(result) {
        $('.count', element).text(result.count);
    }

    var handlerUrl = runtime.handlerUrl(element, 'increment_count')

    $('p', element).click(function(eventObject) {
        $.ajax({
            type: "POST",
            url: handlerUrl,
            data: JSON.stringify({"hello": "world"}),
            success: updateCount
        });
    });
```

XBlock Fields

- Boolean
- Dict
- Float
- Integer
- List
- Set
- String
- XMLString

- ✓ scope=Scope.user_state
- ✓ default=…
- ✓ values={"min": 0, "max": 10}
- □ display_name="Display Name”
- □ help=“Help message"
- □ multiline_editor=True
- □ resettable_editor=False

XBlock Field Scopes

Block Scope User Scope	Block usage (one run of a course)	Block definition	Block type	All blocks
No user	Scope.settings (e.g. deadlines)	**Scope.content** (all course runs)		
One user	**Scope.user_state**		Scope.preferences	Scope.user_info (e.g. timezone)
All user	**Scope.user_state_summary**			

XBlock View

- student_view
- studio_view
  - xblockutils.studio_editable.StudioEditableXBlockMixin
    - editable_fields
    - validate_field_data
- author_view
- fallback_view

Xblock Fragments

- frag = Fragment(html_str.format(self=self))
- frag.add_css
- frag.add_css_url
  - self.runtime.local_resource_url(self, "public/style.css")
- frag.add_javascript
- frag.add_javascript_url
- frag.add_resource
- frag.add_resource_url
- frag.initialize_js("GoQuizXBlock")

XBlock Events & Grading

- has_score
- max_score
- xblock.scorable.ScorableXBlockMixin
  - get_score
  - set_score
  - calculate_score
  - max_score
  - publish

- self.runtime.publish(self, " grade ", {" value " : 60, "max_value": 100 })

XBlock Runtime

1. Field access
2. HTML rendering
3. JavaScript to DOM
4. Route http handlers

- E.g. xblock-sdk, edx lms/cms

XBlock API

https://edx.readthedocs.io/projects/xblock/en/latest/xblock.html

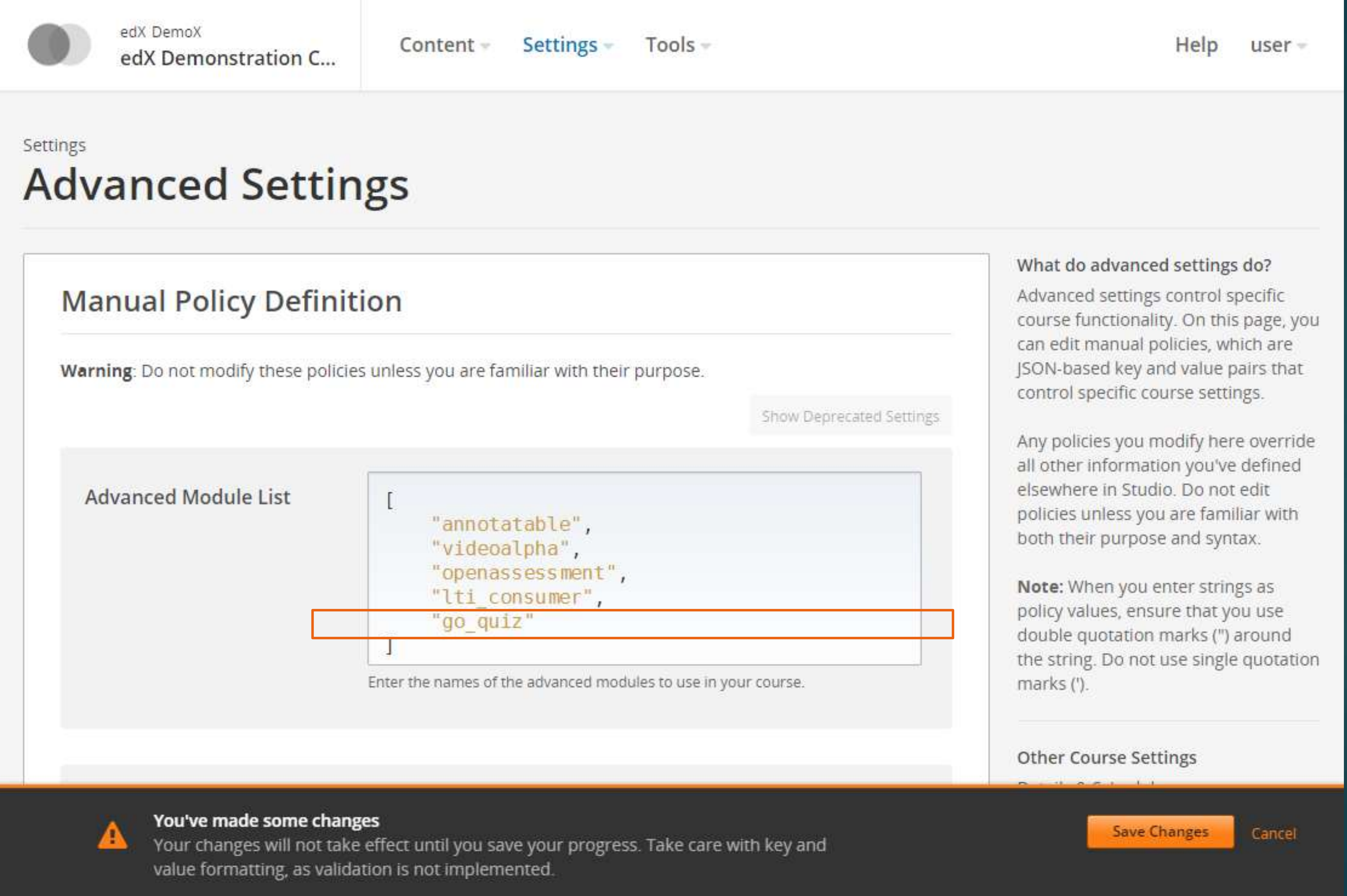
- xblock.core
  - *get_children, get_child, get_parent*
- xblock.fields
- xblock.runtime
- xblock.fragment
- xblock.exceptions
- https://github.com/edx/i18n-tools

Install to Open edX

- (venv) $ pip install /path/to/go_quiz
- (venv) $ ./manage.py lms --settings=production collectstatic --noinput
- (venv) $ ./manage.py cms --settings=production collectstatic --noinput

- display_name
- icon_class
  - ✓ other
  - ✓ video
  - ✓ problem

Advanced Module List



Settings

Advanced Settings

Manual Policy Definition

Warning: Do not modify these policies unless you are familiar with their purpose.

Advanced Module List

```
[
    "annotatable",
    "videoalpha",
    "openassessment",
    "lti_consumer",
    "go_quiz"
]
```

Enter the names of the advanced modules to use in your course.

What do advanced settings do?

Advanced settings control specific course functionality. On this page, you can edit manual policies, which are JSON-based key and value pairs that control specific course settings.

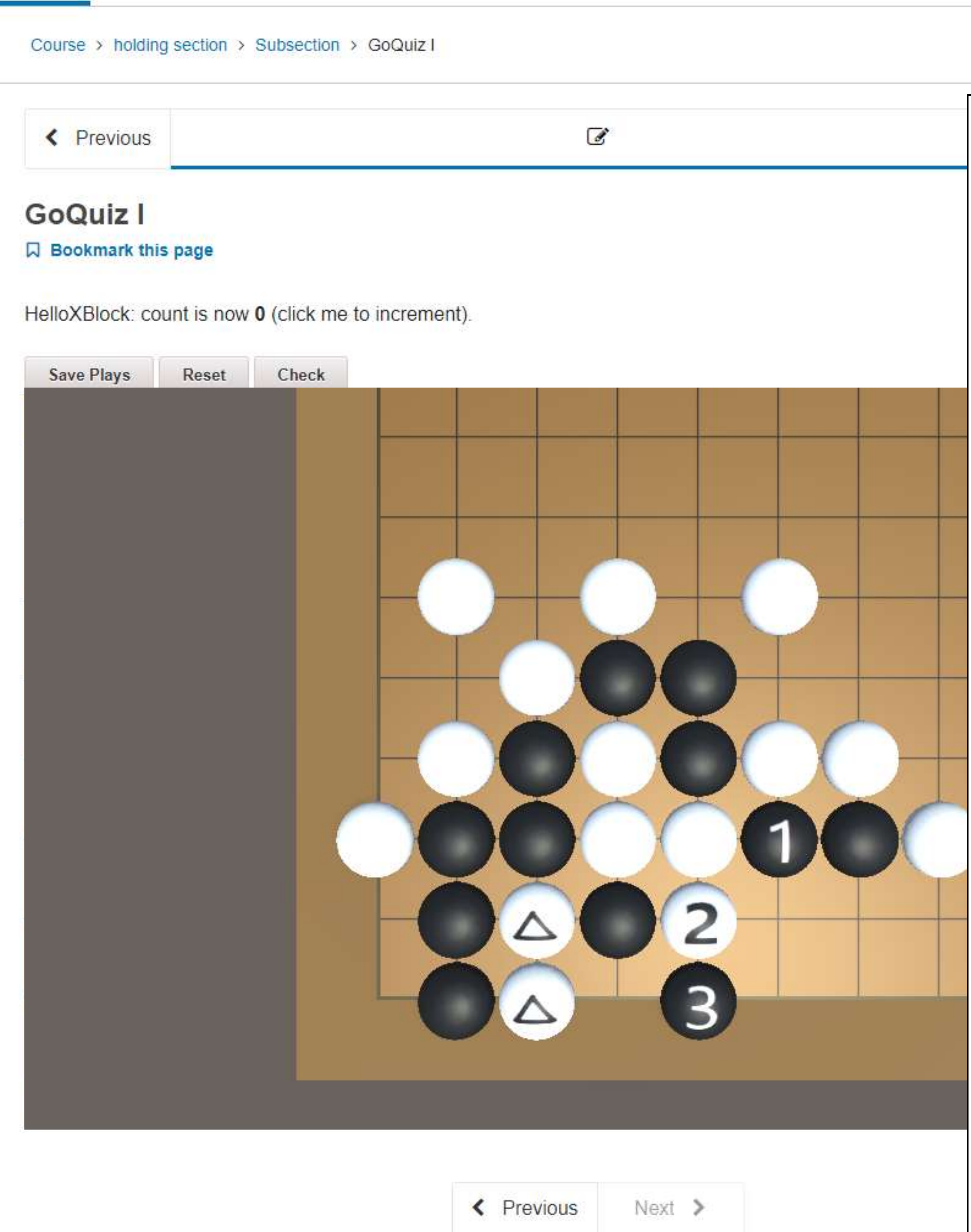
Any policies you modify here override all other information you've defined elsewhere in Studio. Do not edit policies unless you are familiar with both their purpose and syntax.

Note: When you enter strings as policy values, ensure that you use double quotation marks (") around the string. Do not use single quotation marks (').

Other Course Settings

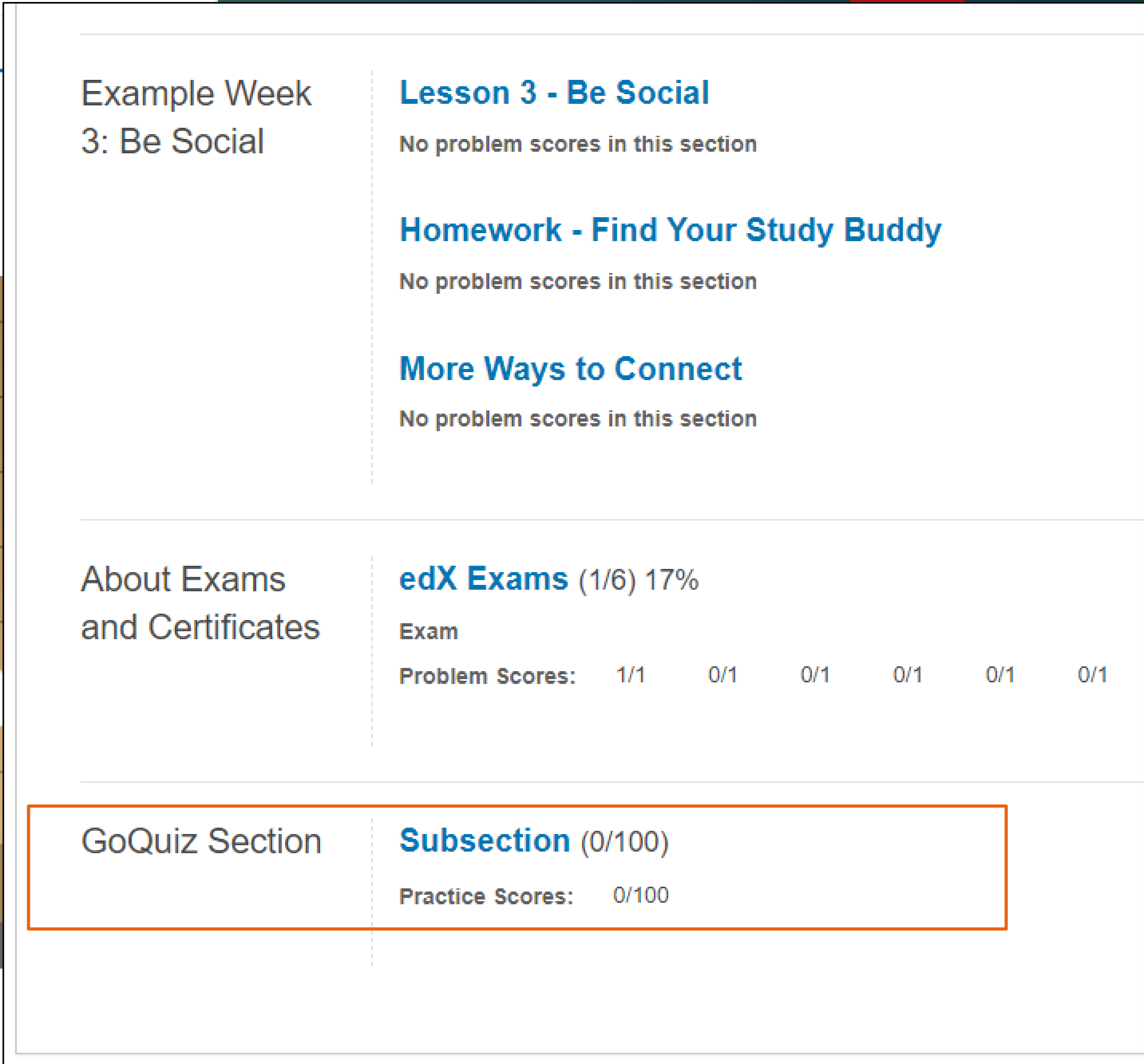
You've made some changes

Your changes will not take effect until you save your progress. Take care with key and value formatting, as validation is not implemented.



GoQuiz I

HelloXBlock: count is now **0** (click me to increment).



Example Week 3: Be Social

Lesson 3 - Be Social

No problem scores in this section

Homework - Find Your Study Buddy

No problem scores in this section

More Ways to Connect

No problem scores in this section

About Exams and Certificates

edX Exams (1/6) 17%

Exam

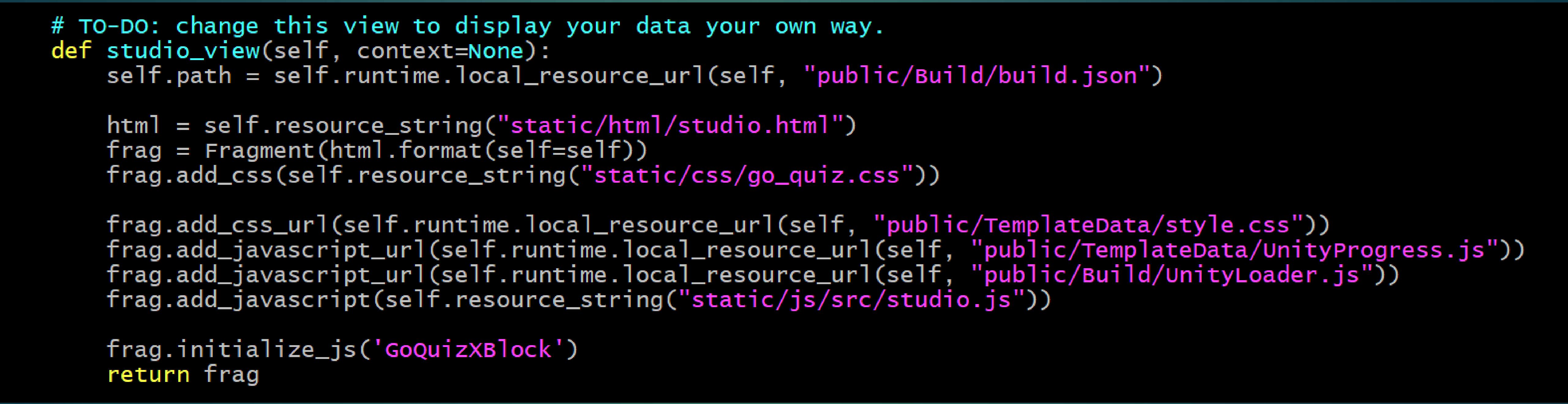
Problem Scores: 1/1 0/1 0/1 0/1 0/1 0/1

GoQuiz Section

Subsection (0/100)

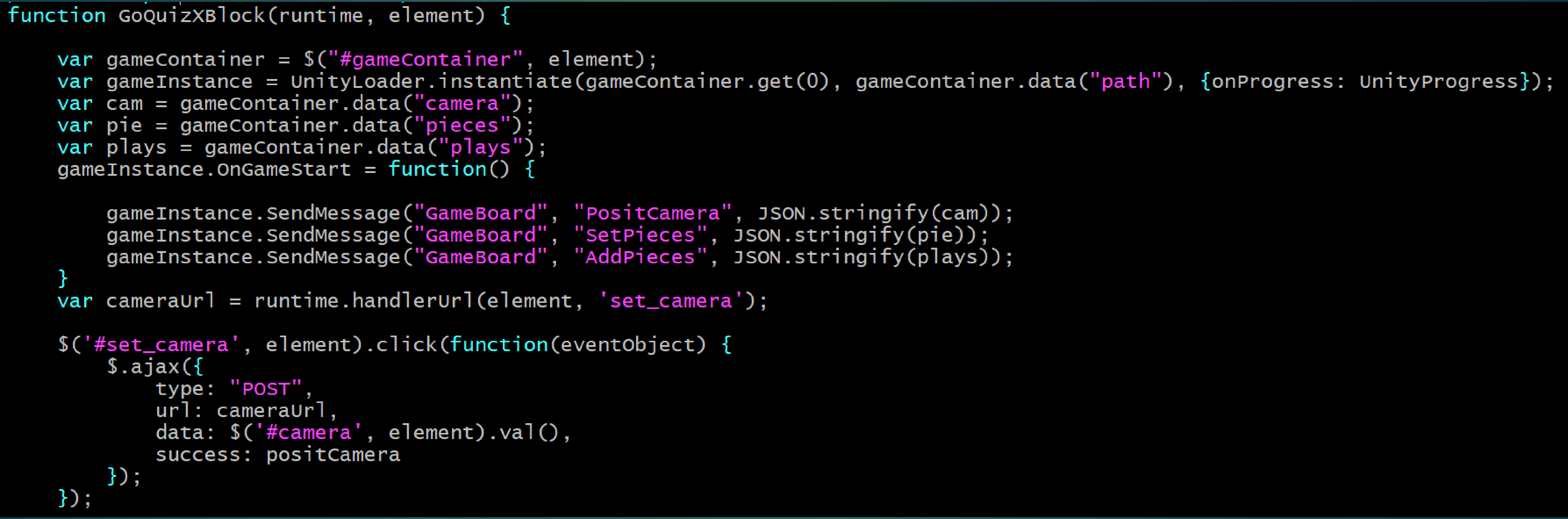
Practice Scores: 0/100

studio_view



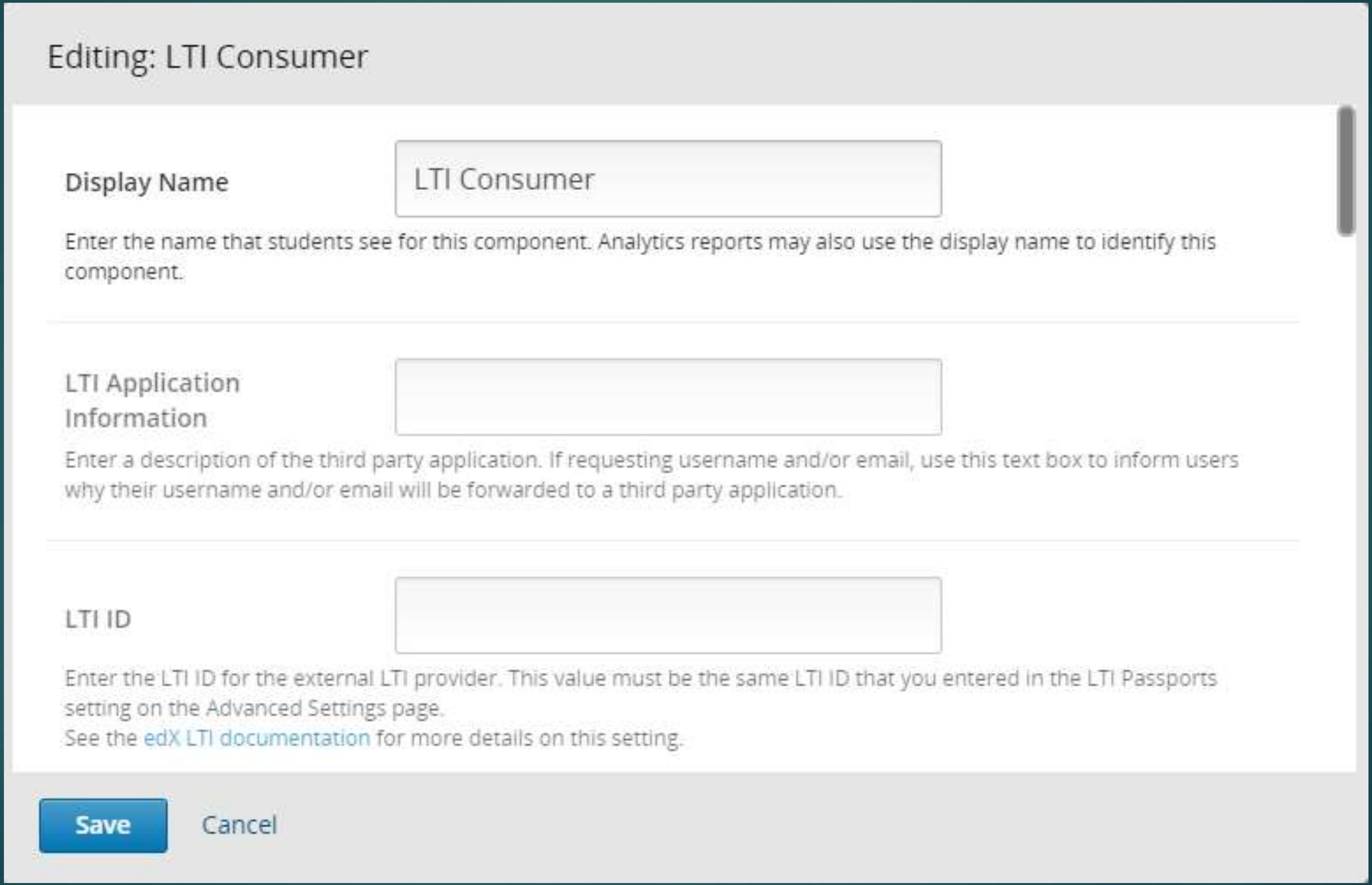
```
# TO-DO: change this view to display your data your own way.
def studio_view(self, context=None):
    self.path = self.runtime.local_resource_url(self, "public/Build/build.json")

    html = self.resource_string("static/html/studio.html")
    frag = Fragment(html.format(self=self))
    frag.add_css(self.resource_string("static/css/go_quiz.css"))

    frag.add_css_url(self.runtime.local_resource_url(self, "public/TemplateData/style.css"))
    frag.add_javascript_url(self.runtime.local_resource_url(self, "public/TemplateData/UnityProgress.js"))
    frag.add_javascript_url(self.runtime.local_resource_url(self, "public/Build/UnityLoader.js"))
    frag.add_javascript(self.resource_string("static/js/src/studio.js"))

    frag.initialize_js('GoQuizXBlock')
    return frag
```

studio.js



```
function GoQuizXBlock(runtime, element) {

    var gameContainer = $("#gameContainer", element);
    var gameInstance = UnityLoader.instantiate(gameContainer.get(0), gameContainer.data("path"), {onProgress: UnityProgress});
    var cam = gameContainer.data("camera");
    var pie = gameContainer.data("pieces");
    var plays = gameContainer.data("plays");
    gameInstance.OnGameStart = function() {

        gameInstance.SendMessage("GameBoard", "PositCamera", JSON.stringify(cam));
        gameInstance.SendMessage("GameBoard", "SetPieces", JSON.stringify(pie));
        gameInstance.SendMessage("GameBoard", "AddPieces", JSON.stringify(plays));
    }
    var cameraUrl = runtime.handlerUrl(element, 'set_camera');

    $('#set_camera', element).click(function(eventObject) {
        $.ajax({
            type: "POST",
            url: cameraUrl,
            data: $('#camera', element).val(),
            success: positCamera
        });
    });
```

lti_consumer



Editing: LTI Consumer

Display Name: LTI Consumer

Enter the name that students see for this component. Analytics reports may also use the display name to identify this component.

LTI Application Information:

Enter a description of the third party application. If requesting username and/or email, use this text box to inform users why their username and/or email will be forwarded to a third party application.

LTI ID:

Enter the LTI ID for the external LTI provider. This value must be the same LTI ID that you entered in the LTI Passports setting on the Advanced Settings page.
See the edX LTI documentation for more details on this setting.