```
/*
 * Date Created: Monday, November 22, 2021 7:27 AM
 * Author: enaielei <nommel.isanar.lavapie.amolat@gmail.com>
 *
 * Copyright © 2021 CoDe_A. All Rights Reserved.
 */

using System;
using System.Collections.Generic;
using System.Linq;
using TMPro;
using UnityEngine;
using UnityEngine.UI;
using Utilities;

namespace Ph.CoDe_A.Lakbay.Core
{
    using Layout = Tuple<Entry.Type, LayoutGroup>;

    public class Content : Controller
    {
        protected IEnumerable<string> _oldValues;
        protected virtual IEnumerable<string> _values =>
            content?.Select((e) => e.ToString());

        [Serializable]
        public struct Group<T0, T1>
            where T0 : Component
        {
            public LayoutGroup layout;
            public T0 component;
            [SerializeField]
            private Viewer<T0, T1> _viewer;
            public Viewer<T0, T1> viewer
            {
                get => _viewer ?? FindObjectOfType<Viewer<T0, T1>>();
                set => _viewer = value;
            }
        }

        protected Layout _previousLayout;

        public bool automatic = true;
        public LayoutGroup root;
        public Group<TextMeshProUGUI, string> textGroup;
        public Group<Image, Sprite> imageGroup;
        public List<Entry> content = new List<Entry>();

        public override void Awake()
        {
            base.Awake();
        }

        public override void Update()
        {
            base.Update();
            if (automatic)
            {
                var values = _values.ToArray();
                if (_oldValues == null || _oldValues.Count() == 0
```

```
            || _oldValues.Enumerate().Any((h) => values[h.Key] != h.Value))
        {
            _oldValues = values;
            Build();
        }
    }
}

[ContextMenu("Clear")]
public virtual void Clear()
{
    if (!root) return;
    if (Application.isPlaying) this.root.transform.DestroyChildren();
    else this.root.transform.DestroyChildrenImmediately();
    _previousLayout = null;
}

[ContextMenu("Build")]
public virtual void Build() => Build(content);

public virtual void Build(IEnumerable<Entry> content) =>
    Build(content.ToArray());

public virtual void Build(params Entry[] content)
{
    if (root)
    {
        Clear();
        this.content.Clear();
        this.content.AddRange(content);
        var root = this.root;

        foreach (var entry in content)
        {
            switch (entry.type)
            {
                case Entry.Type.Text:
                    Build(entry, textGroup,
                        (c) => c.SetText(entry.text.value),
                        (v) => v.Show(entry.text.value));
                    break;
                case Entry.Type.Image:
                    Build(entry, imageGroup,
                        (c) => c.sprite = entry.GetAsset<Sprite>(),
                        (v) =>
                        {
                            (v as ImageViewer).Show(
                                entry.GetAsset<Sprite>(),
                                entry.image.description,
                                entry.image.source
                            );
                        }
                    );
                    break;
                default: break;
            }
        }
    }
}
```

```
        public virtual T0 Build<T0, T1>(
            Entry entry, Group<T0, T1> group,
            Action<T0> onComponentBuild = default,
            Action<Viewer<T0, T1>> onViewerBuild = default)
            where T0 : Component
        {
            T0 component = default;
            var root = this.root;

            if (_previousLayout == null || _previousLayout.Item1 != entry.type)
            {
                root = group.layout ? Instantiate(group.layout, root.transform)
                    : root;
                _previousLayout = new Layout(entry.type, root);
            }
            else if (_previousLayout != null && _previousLayout.Item1 == entry.type)
            {
                root = _previousLayout.Item2;
            }

            if (group.component)
            {
                component =
                    Instantiate(group.component, root.transform);
                onComponentBuild?.Invoke(component);
            }

            if (group.viewer && component)
            {
                var button = component.gameObject.EnsureComponent<Button>();
                button.onClick.AddListener(
                    () => onViewerBuild?.Invoke(group.viewer));
            }

            return component;
        }

        public virtual void Build(TextAsset asset)
        {
            if (asset) Build(asset.text.DeserializeAsYaml<List<Entry>>());
        }
    }
}
```