FRIB logo application based on Qt Design 3D animation

Sasha Tarasova
Research Assistant

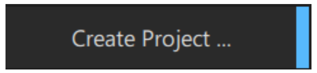
tarasoal@frib.msu.edu

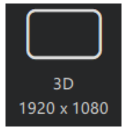
28 March 2024

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics and used resources of the Facility for Rare Isotope Beams (FRIB) Operations, which is a DOE Office of Science User Facility under Award Number DE-SC0023633, and by the US National Science Foundation under Grants No. PHY-20-12040 and 23-10078 “Windows on the Universe: Open Quantum Systems in Atomic Nuclei at FRIB”.

3D Animation Setup

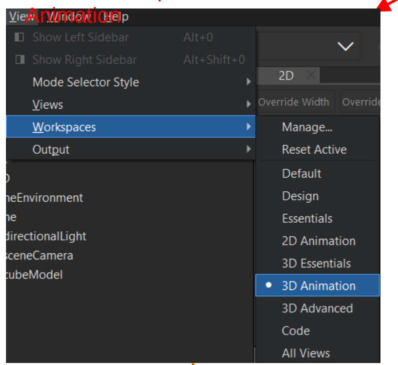
1. Open QT Design Studio
2. Select Create Project… 3D

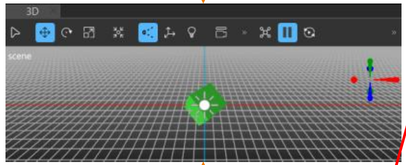


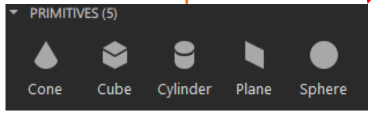


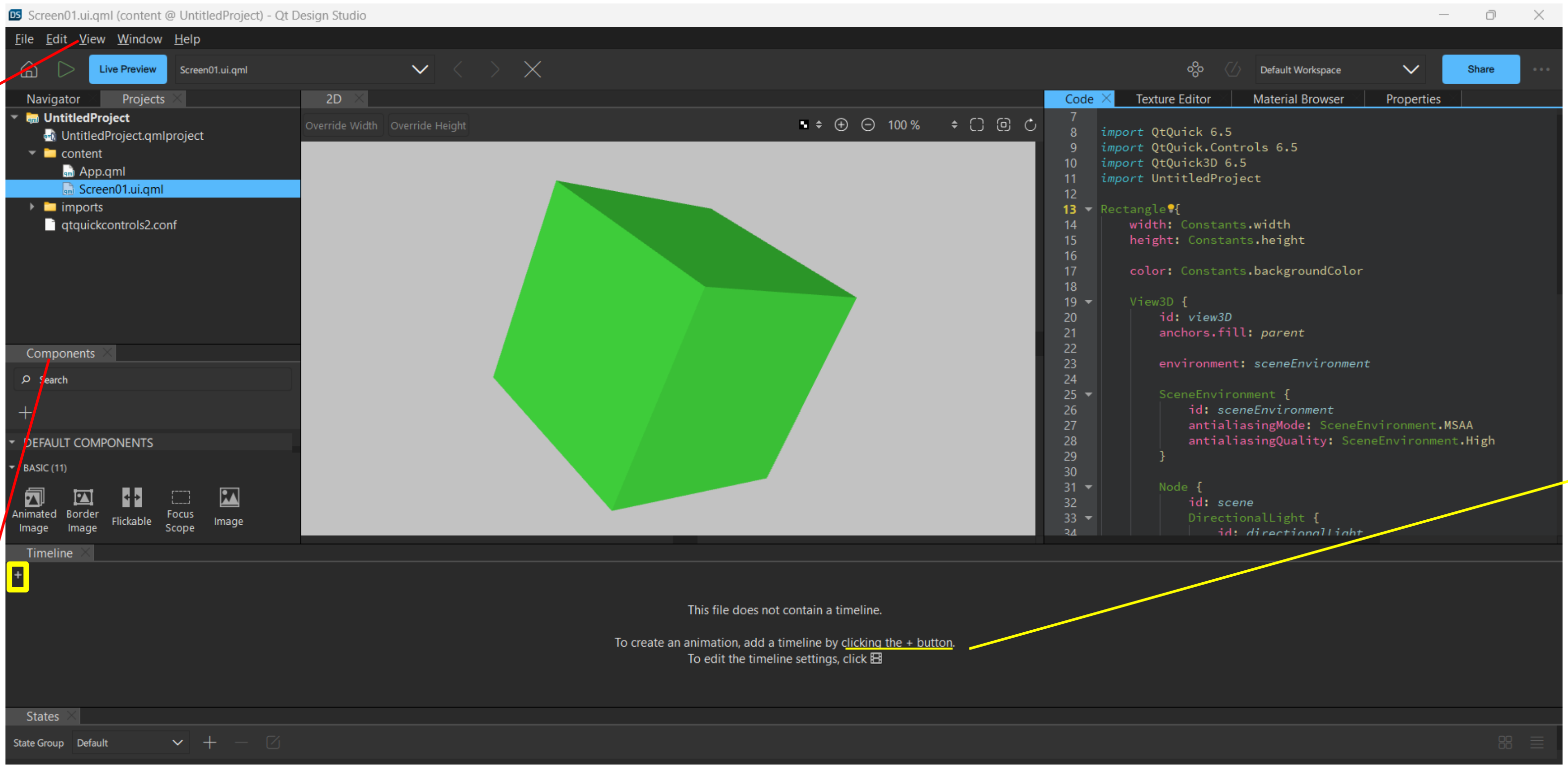
- Add 3D space

View > Workspaces > 3D Animation









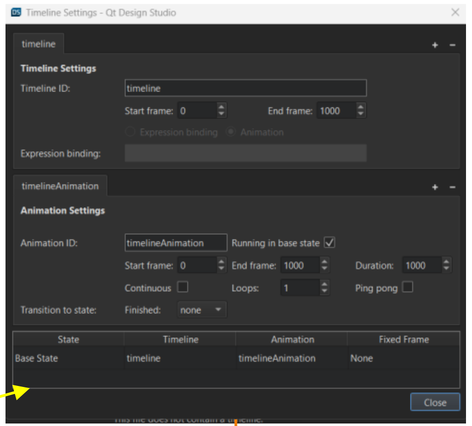
```
import QtQuick 6.5
import QtQuick.Controls 6.5
import QtQuick3D 6.5
import UntitledProject

Rectangle {
    width: Constants.width
    height: Constants.height

    color: Constants.backgroundColor

    View3D {
        id: view3D
        anchors.fill: parent

        environment: sceneEnvironment

        SceneEnvironment {
            id: sceneEnvironment
            antialiasingMode: SceneEnvironment.MSAA
            antialiasingQuality: SceneEnvironment.High
        }

        Node {
            id: scene
            DirectionalLight {
                id: directionalLight
```

This file does not contain a timeline.

To create an animation, add a timeline by clicking the + button.
To edit the timeline settings, click

Drag Primitives into 3D space to manipulate them

- Add Timeline



timeline

Timeline Settings

Timeline ID: timeline

Start frame: 0 End frame: 1000

Expression binding / Animation

Expression binding:

timelineAnimation

Animation Settings

Animation ID: timelineAnimation Running in base state ✓

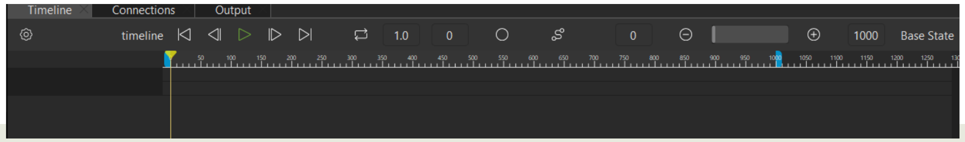
Start frame: 0 End frame: 1000 Duration: 1000

Continuous ☐ Loops: 1 Ping pong ☐

Transition to state: Finished: none

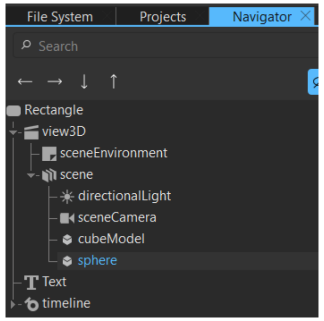
State	Timeline	Animation	Fixed Frame
Base State	timeline	timelineAnimation	None

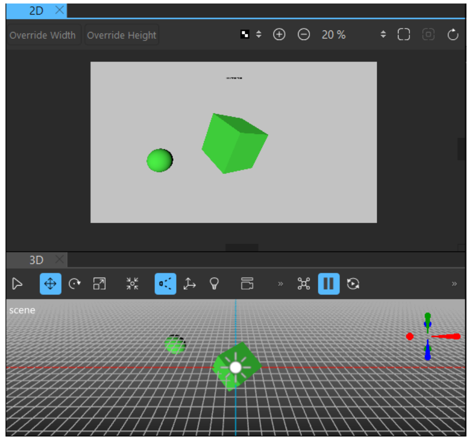
Adjust animation frames, duration, and transitions



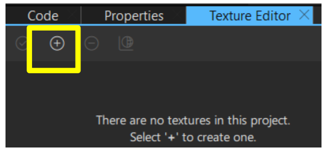
Displaying Images on Objects

1. Select object of choice from the Navigator and go to the Texture Editor on right-hand side

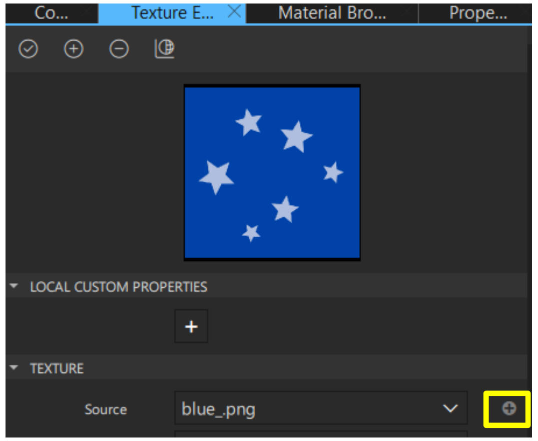




2. Create a new texture and select a file for the source



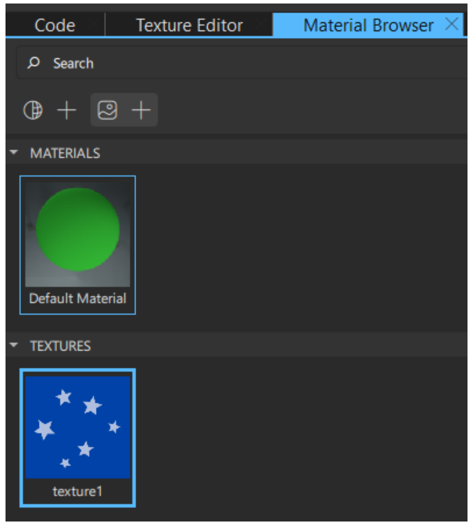
There are no textures in this project.
Select '+' to create one.



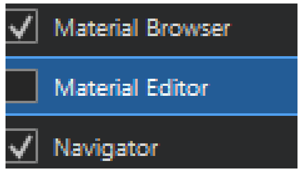
Source: blue_.png

3. Drag the texture and drop onto a material

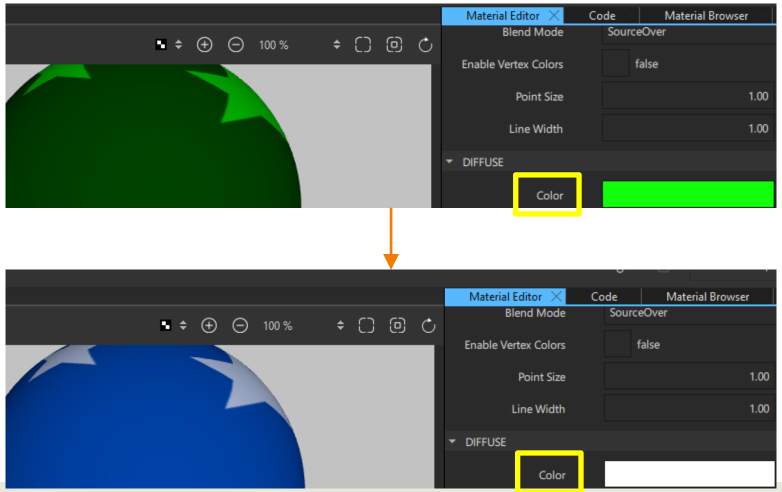
Add additional materials and textures



4. Edit the material by going into View > Material Editor



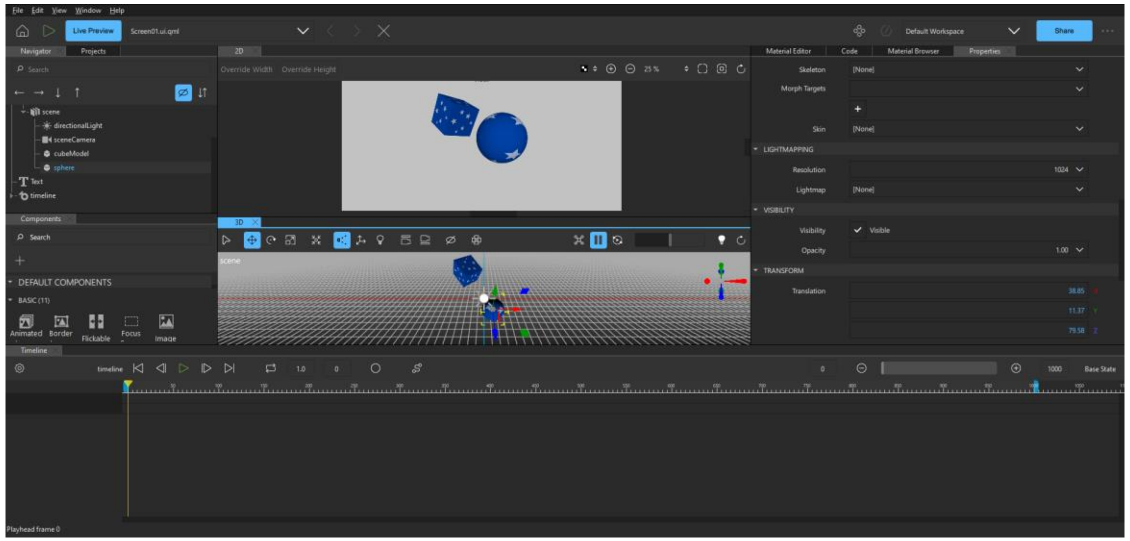
Tip: If you hit Apply on the "Select a material property" pop-up immediately, it goes the texture goes into a "diffuseMap" material. To see the correct color, switch the color to white under Material Editor > Diffuse > Color



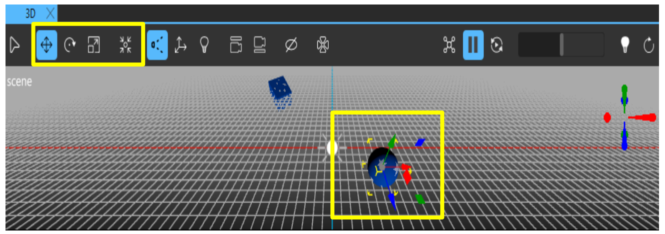
Blend Mode: SourceOver
Enable Vertex Colors: false
Point Size: 1.00
Line Width: 1.00

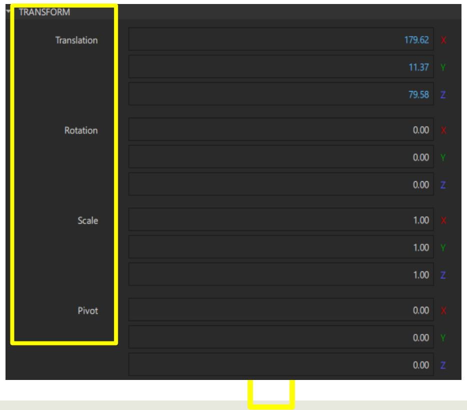
Creating Animation Using a Timeline

1. Make sure your workspace is set up so you can see the 2D and 3D views, the Timeline, and the Properties Tab

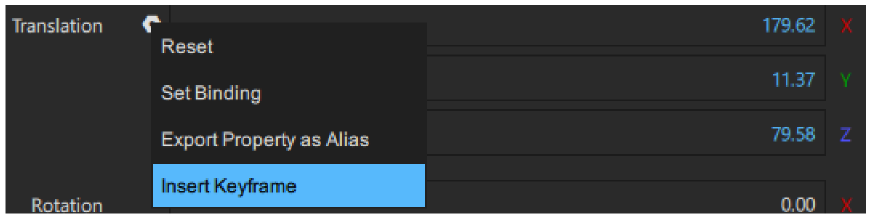


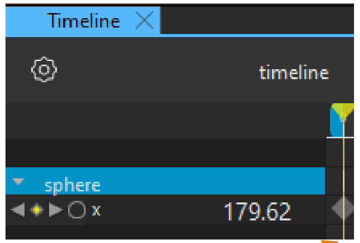
2. Transform an object clicking on it in the 3D view and using the tools, or through the Transform tab



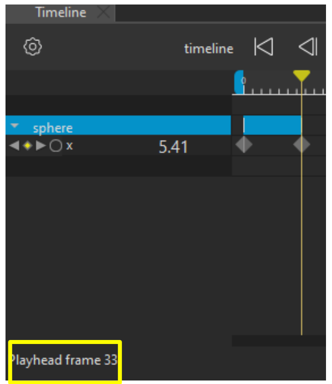

TRANSFORM
Translation 179.62 X 11.37 Y 79.58 Z
Rotation 0.00 X 0.00 Y 0.00 Z
Scale 1.00 X 1.00 Y 1.00 Z
Pivot 0.00 X 0.00 Y 0.00 Z

3. Once you have set your starting position, click on this icon: and select "Insert Keyframe" to add that property to the timeline





4. Drag the yellow bar to the frame where you want your next change to happen, change a property, and insert a new keyframe

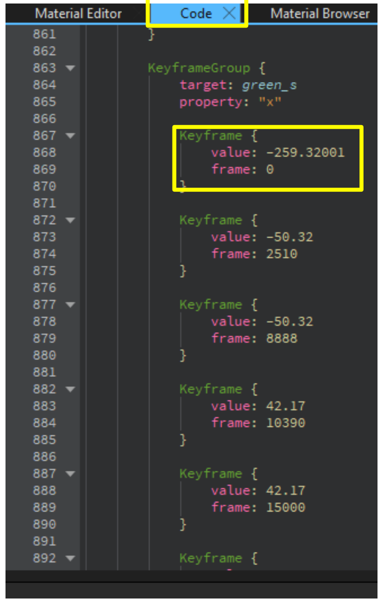


5. Drag the yellow bar across the blue timeline, or hit these buttons: to see your changes in the 2D View

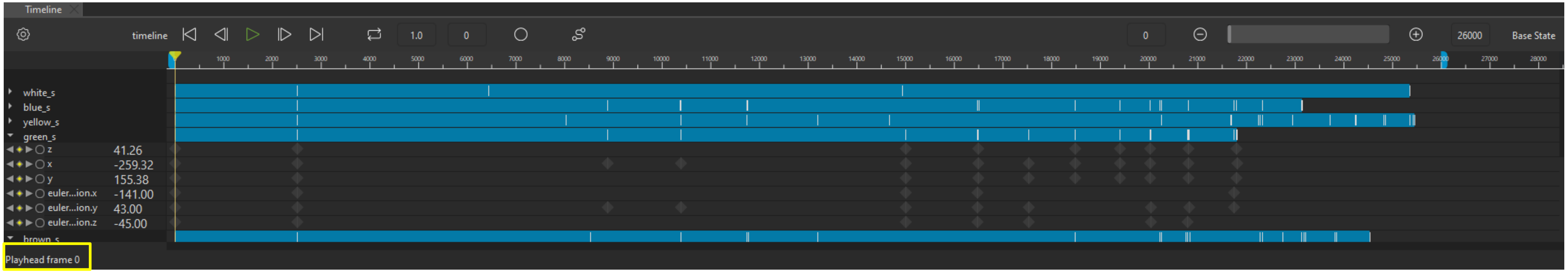
Important

1. Make sure you save the project often through the File tab, as well as multiple versions, as the program can crash unexpectedly!

2. To have multiple transformations at once, you need to make sure you insert a keyframe for <u>the start and end</u> of **each** property change. You can use the Code tab, or drag the yellow bar, to find the specific frame you are looking for



```
    }

    KeyframeGroup {
        target: green_s
        property: "x"

        Keyframe {
            value: -259.32001
            frame: 0
        }

        Keyframe {
            value: -50.32
            frame: 2510
        }

        Keyframe {
            value: -50.32
            frame: 8888
        }

        Keyframe {
            value: 42.17
            frame: 10390
        }

        Keyframe {
            value: 42.17
            frame: 15000
        }

        Keyframe {
```

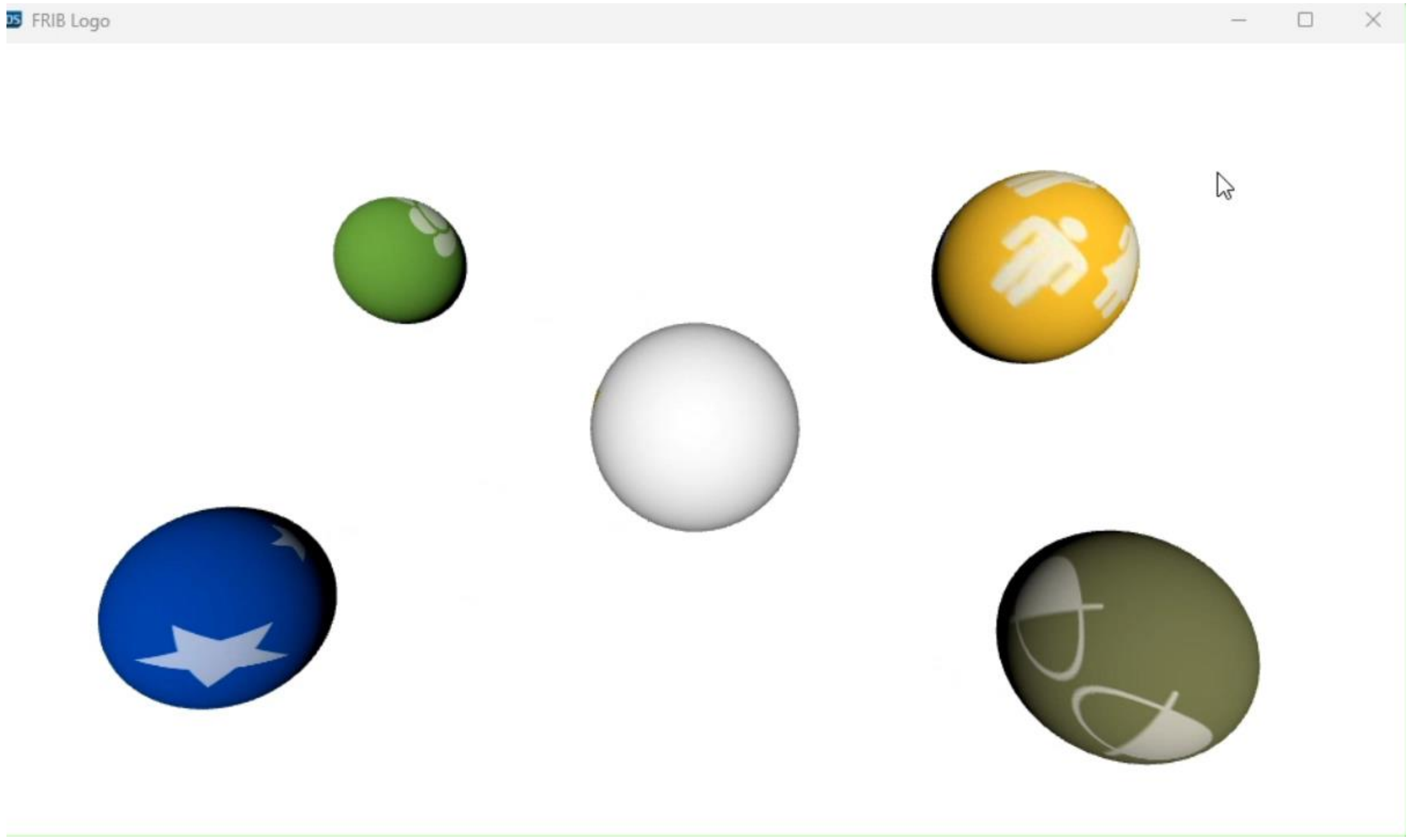


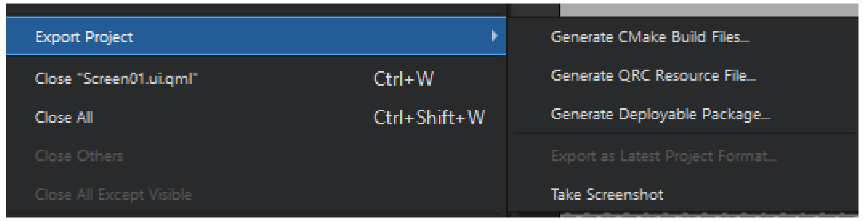
```
z             41.26
x            -259.32
y             155.38
euler...ion.x -141.00
euler...ion.y  43.00
euler...ion.z -45.00
```

Playhead frame 0

Finishing Up

Once you are happy with your project, make sure to save and export the project to the necessary files with File → Export Project

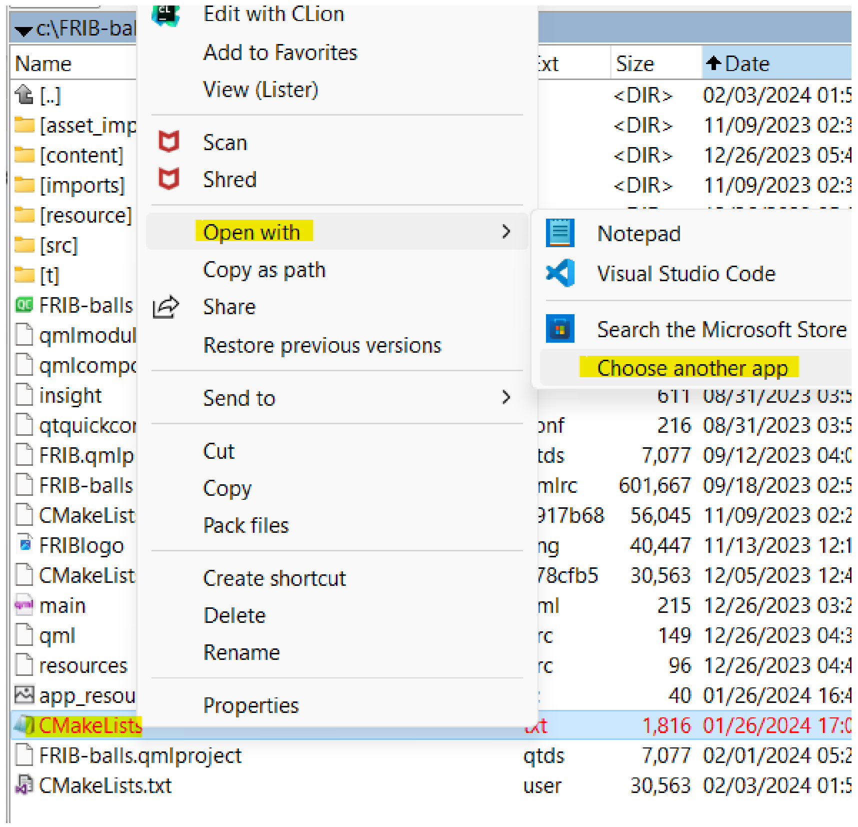



Export Project
Close "Screen01.ui.qml" Ctrl+W
Close All Ctrl+Shift+W
Close Others
Close All Except Visible
Generate CMake Build Files...
Generate QRC Resource File...
Generate Deployable Package...
Export as Latest Project Format...
Take Screenshot

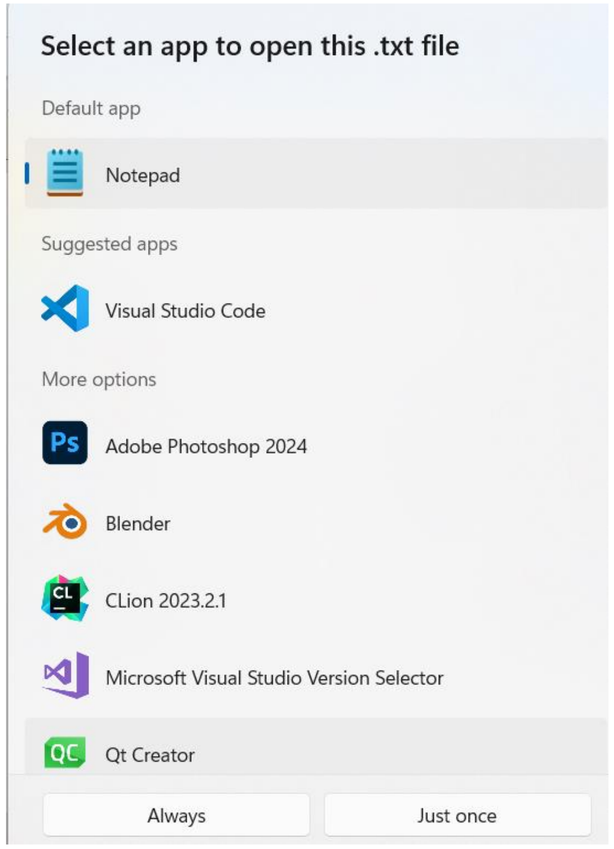
You can also change the speed with the start and end frames, and whether you want a continuous, or even ping pong reverse animation, through the timeline settings

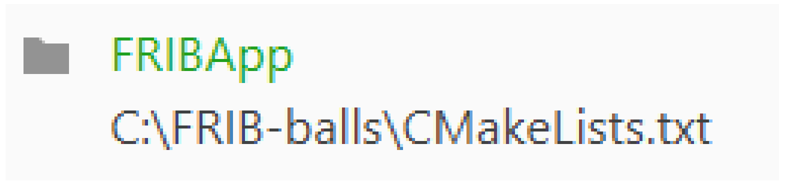
Exporting to Qt Creator

To export to Qt Creator, open the file “CMakeLists.txt” with Qt Creator



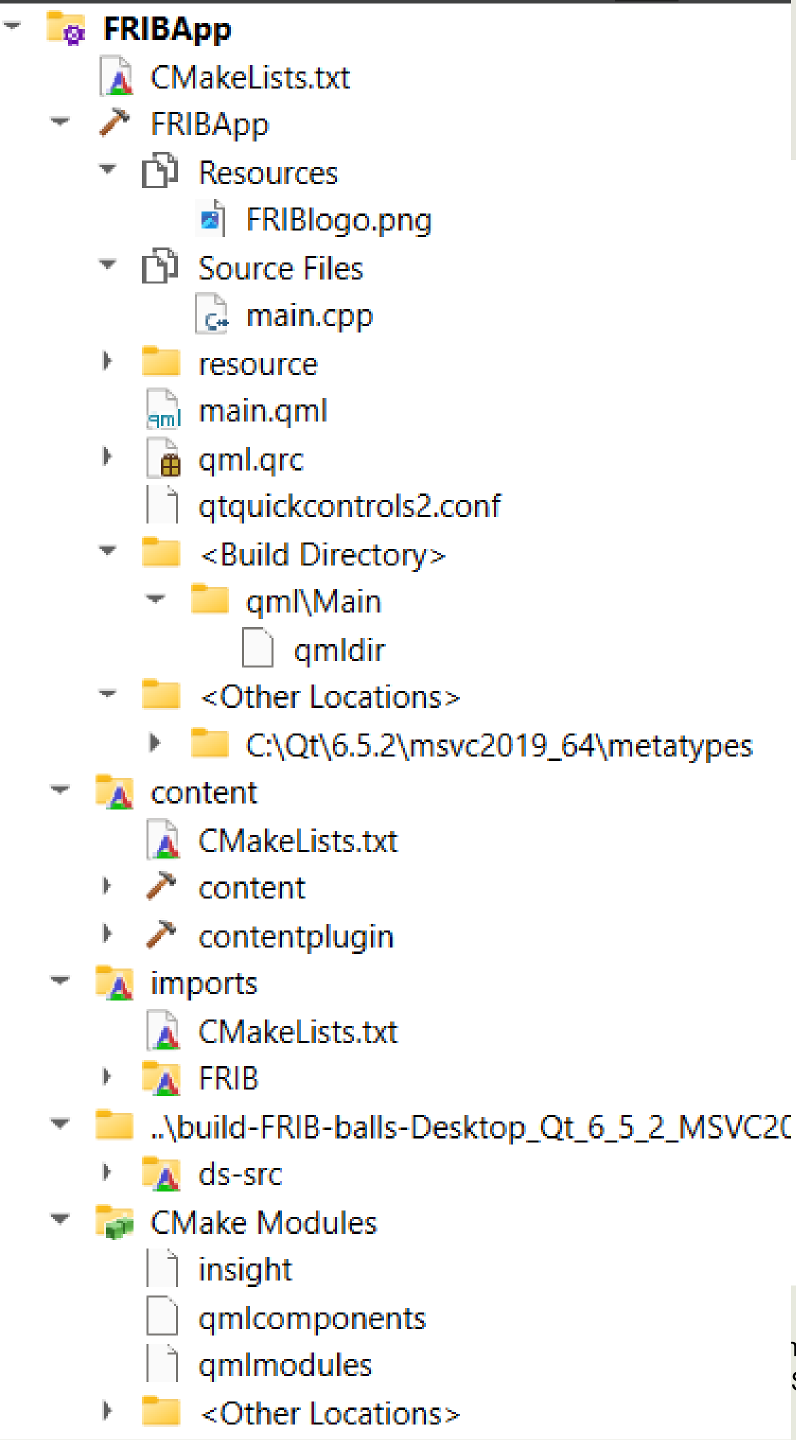
Ext	Size	Date
	<DIR>	02/03/2024 01:5
	<DIR>	11/09/2023 02:3
	<DIR>	12/26/2023 05:4
	<DIR>	11/09/2023 02:3
	611	08/31/2023 03:5
onf	216	08/31/2023 03:5
tds	7,077	09/12/2023 04:0
mlrc	601,667	09/18/2023 02:5
917b68	56,045	11/09/2023 02:2
ng	40,447	11/13/2023 12:1
78cfb5	30,563	12/05/2023 12:4
ml	215	12/26/2023 03:2
rc	149	12/26/2023 04:3
rc	96	12/26/2023 04:4
	40	01/26/2024 16:4
xt	1,816	01/26/2024 17:0
qtds	7,077	02/01/2024 05:2
user	30,563	02/03/2024 01:5



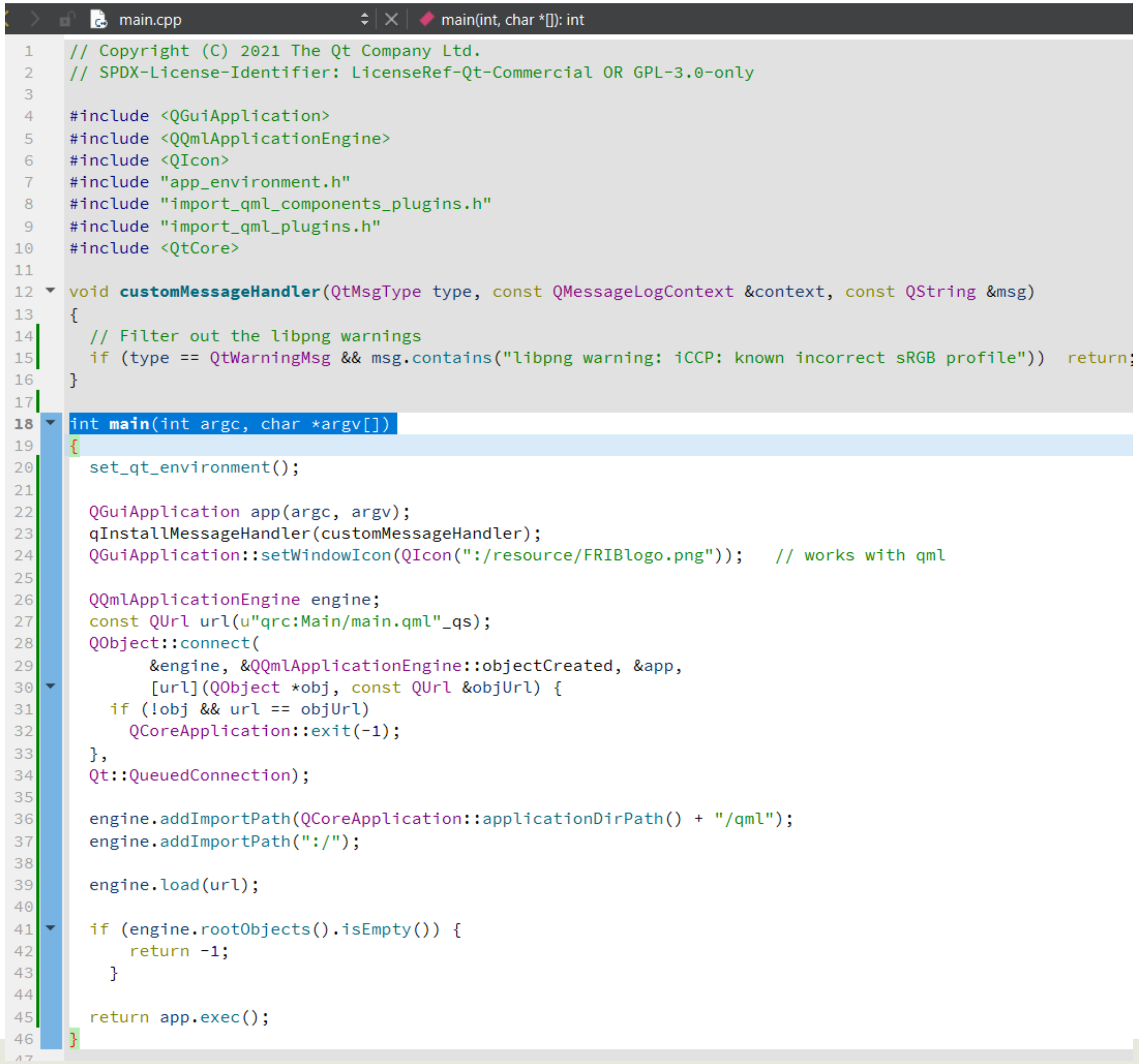


To run the animation through Qt Creator, make sure you have all the necessary libraries and resources installed!

Qt Creator : FRIBapp project



```
FRIBApp
    CMakeLists.txt
    FRIBApp
        Resources
            FRIBlogo.png
        Source Files
            main.cpp
        resource
        main.qml
        qml.qrc
        qtquickcontrols2.conf
        <Build Directory>
            qml\Main
                qmldir
        <Other Locations>
            C:\Qt\6.5.2\msvc2019_64\metatypes
    content
        CMakeLists.txt
        content
        contentplugin
    imports
        CMakeLists.txt
        FRIB
    ..\build-FRIB-balls-Desktop_Qt_6_5_2_MSVC20
        ds-src
    CMake Modules
        insight
        qmlcomponents
        qmlmodules
        <Other Locations>
```



```cpp
// Copyright (C) 2021 The Qt Company Ltd.
// SPDX-License-Identifier: LicenseRef-Qt-Commercial OR GPL-3.0-only

#include <QGuiApplication>
#include <QQmlApplicationEngine>
#include <QIcon>
#include "app_environment.h"
#include "import_qml_components_plugins.h"
#include "import_qml_plugins.h"
#include <QtCore>

void customMessageHandler(QtMsgType type, const QMessageLogContext &context, const QString &msg)
{
  // Filter out the libpng warnings
  if (type == QtWarningMsg && msg.contains("libpng warning: iCCP: known incorrect sRGB profile"))  return;
}

int main(int argc, char *argv[])
{
  set_qt_environment();

  QGuiApplication app(argc, argv);
  qInstallMessageHandler(customMessageHandler);
  QGuiApplication::setWindowIcon(QIcon(":/resource/FRIBlogo.png"));   // works with qml

  QQmlApplicationEngine engine;
  const QUrl url(u"qrc:Main/main.qml"_qs);
  QObject::connect(
        &engine, &QQmlApplicationEngine::objectCreated, &app,
        [url](QObject *obj, const QUrl &objUrl) {
    if (!obj && url == objUrl)
      QCoreApplication::exit(-1);
  },
  Qt::QueuedConnection);

  engine.addImportPath(QCoreApplication::applicationDirPath() + "/qml");
  engine.addImportPath(":/");

  engine.load(url);

  if (engine.rootObjects().isEmpty()) {
      return -1;
    }

  return app.exec();
}
```

Animated gif after the code video-capture