Fiji ImageJ
Sum Projection Macro Code

SumProjection.ijm

```
// Macro to create a sum orthogonal projection of directory of images

// The next line prevents ImageJ from showing the processing steps during
// processing of a large number of images, speeding up the macro
//setBatchMode(true);

// Show the user a dialog to select a directory of images
inputDirectory = getDirectory("Choose a Directory of Images");

// Get the list of files from that directory
// NOTE: if there are non-image files in this directory, it may cause the macro to crash
fileList = getFileList(inputDirectory);

for (i = 0; i < fileList.length; i++)
{
    processImage(fileList[i]);
}

//setBatchMode(false); // Now disable BatchMode since we are finished
updateResults();  // Update the results table so it shows the filenames

function processImage(imageFile)
{
    open(imageFile);
    // Get the filename from the title of the image that's open for adding to the results table
    // We do this instead of using the imageFile parameter so that the
    // directory path is not included on the table
    filename = getTitle();

    // Create an orthogonal projection of the sum of all the slices, arrange the channels,
    // and make a composite image
    run("Z Project...", "projection=[Sum Slices]");
    run("Arrange Channels...", "new=21");
    ProjectionName = "SUM_" + filename;

    // Duplicate the projection to create the combo channel, set both channels as cyan,
    // make a composite, and stack to RGB
    run("Duplicate...", "duplicate");
    Stack.setChannel(1);
    run("Cyan");
    Stack.setChannel(2);
    run("Cyan");
    run("Make Composite");
    run("Stack to RGB");
    setOption("ScaleConversions", true);
    run("16-bit");
    CyanChannel = "[" + replace(ProjectionName,".czi","-1.czi (RGB)]");

    // Split channels of original projection
    selectImage(ProjectionName);
    run("Split Channels");
    RedChannel = "C1-" + ProjectionName;
    GreenChannel = "C2-" + ProjectionName;
    selectImage(RedChannel);
    setOption("ScaleConversions", true);
    run("16-bit");
    selectImage(GreenChannel);
    setOption("ScaleConversions", true);
    run("16-bit");
    run("Merge Channels...", "c1=" + RedChannel + " c2=" + GreenChannel + " c5=" + CyanChannel + " create");

    // Save the generated images in the same directory
    SaveName = ProjectionName;
    saveAs("tiff", inputDirectory+SaveName);

    close("*");  // Closes all images
}
```