

# ArtiSynth Coding Standard

---

**John Lloyd**

August 20, 2014

## Contents

|          |   |          |
|----------|---|----------|
| <b>1</b> | <b>Introduction</b>   | <b>3</b> |
| <b>2</b> | <b>Cuddle braces and indent by 3</b>                              | <b>3</b> |
| <b>3</b> | <b>Always used braced blocks</b>                                  | <b>3</b> |
| <b>4</b> | <b>Do not use tabs</b>  | <b>3</b> |
| <b>5</b> | <b>Keep line widths to 80 columns</b>                             | <b>4</b> |
| <b>6</b> | <b>Break lines at the beginning of argument lists</b>             | <b>4</b> |
| <b>7</b> | <b>Break lines after assignment operators</b>                     | <b>4</b> |
| <b>8</b> | <b>Align conditional expressions with the opening parentheses</b> | <b>4</b> |
| <b>9</b> | <b>No space before empty argument lists</b>                       | <b>5</b> |

---

## 1 Introduction

It is notoriously difficult to create a coding standard that satisfies all developers, let alone create one that is actually adhered to. Nonetheless, there is an ArtiSynth coding standard which all contributors are encouraged to follow.

The standard is similar in appearance to the [coding rules provided by Sun](#), although it does differ slightly. For users of Eclipse, there is a code style file located in

```
$ARTISYNTH_HOME/support/eclipse/artisynthCodeFormat.xml
```

that supports most of the conventions, though not all, as noted below.

## 2 Cuddle braces and indent by 3

As with the Sun coding rules, all braces are cuddled. Basic indentation is set to 3 to keep code from creeping too far across that page. For example:

```
public static RootModel getRootModel() {
    if (myWorkspace != null) {
        return myWorkspace.getRootModel();
    }
    else {
        return null;
    }
}
```

## 3 Always used braced blocks

As in the Sun rules, code following control constructs is always enclosed in a braced block, even when this is not necessary. That means that

```
for (int i=0; i<cnt; i++) {
    if (i % 2 == 0) {
        System.out.println ("Even");
    }
    else {
        System.out.println ("Odd");
    }
}
```

should be used instead of

```
for (int i=0; i<cnt; i++)
    if (i % 2 == 0)
        System.out.println ("Even");
    else
        System.out.println ("Odd");
```

The reason for this is to provide greater uniformity and to make it easier to add and remove statements from the blocks.

## 4 Do not use tabs

Code should be indented with spaces only. Tabs should not be used since the spacing associated with tabs varies too much between coding environments and can not always be controlled.

## 5 Keep line widths to 80 columns

Again, as with the Sun rules, code should be kept to 80 columns. The idea here is that it is easier to read code that doesn't creep too far across the page. However, to maintain an 80 column width, it will often be necessary to wrap lines.

## 6 Break lines at the beginning of argument lists

If breaking a line is necessary, the beginning of an argument list is a good place to do it. The break should be indented by the usual 3 spaces. The result can look a bit like Lisp:

```
public String getKeyBindings() {
    return artisynth.core.util.TextFromFile.getTextOrError (
        ArtisynthPath.getResource (
            getDirectoryName()+KEY_BINDINGS_FILE));
}
```

If at all possible, do not break lines at the '.' used for method or field references. For clarity, it is better to keep these together with their objects. Therefore, please *do not* do this:

```
public String getKeyBindings() {
    return artisynth.core.util.TextFromFile
    .getTextOrError (ArtisynthPath
    .getResource (getDirectoryName()+KEY_BINDINGS_FILE));
}
```

Note that Eclipse will not generally enforce these breaking conventions, so you need to do this yourself.

## 7 Break lines after assignment operators

Another good place for a line break is after an assignment operator, particularly if the left side is long:

```
LinkedList<ClipPlaneControl> myClipPlaneControls =
    new LinkedList<ClipPlaneControl>();

if (hasMuscle) {
    ComponentList<MuscleBundle> muscles =
        ((MuscleTissue)tissue).getMuscleList();
}
```

Again, Eclipse will not generally enforce this, so it must be done manually.

## 8 Align conditional expressions with the opening parentheses

When line wrapping is used inside a conditional expression, the expression itself should be aligned with the opening parentheses, with operators placed at the right:

```
if (e.getSource() instanceof Rotator3d ||
    e.getSource() instanceof Transrotator3d ||
    e.getSource() instanceof Scaler3d) {
    centerTransform (transform, dragger.getDraggerToWorld());
}
```

Again, note the Eclipse will not generally enforce this, and will instead tend to produce output like this:

```
if (e.getSource() instanceof Rotator3d
|| e.getSource() instanceof Transrotator3d
|| e.getSource() instanceof Scaler3d) {
    centerTransform (transform, dragger.getDraggerToWorld());
}
```

## 9 No space before empty argument lists

No spaces are placed before empty argument lists, as in

```
getMainFrame().getMenuBarHandler().enableShowPlay();

public ViewerManager getViewerManager() {
    return myViewerManager;
}
```

This is largely to improve readability, particularly when accessors are chained together in a single statement.