

# **DaSL Developer Guide**

Scott Chamberlain

# Table of contents

<b>Preface</b>	<b>3</b>
<b>1 Introduction</b>	<b>4</b>
<b>2 Help</b>	<b>5</b>
2.1 House Calls . . . . .	5
2.2 Slack . . . . .	5
<b>3 Contributing</b>	<b>6</b>
3.1 Code of Conduct . . . . .	6
3.2 Communication Channels . . . . .	6
3.3 xxx . . . . .	6
<b>4 Style</b>	<b>7</b>
4.1 Package styler . . . . .	7
4.2 IDE and Text editor support . . . . .	7
<b>5 Code review</b>	<b>8</b>
5.1 DaSL interal guidelines . . . . .	8
5.2 Community guidelines . . . . .	8
<b>6 Package documentation</b>	<b>9</b>
6.1 R . . . . .	9
6.1.1 DaSL pkgdown template . . . . .	9
6.2 Python . . . . .	9
6.3 Guidelines . . . . .	10
<b>References</b>	<b>11</b>

# Preface

This is a Quarto book.

Not sure what to put here. Seems like we have to have a preface for a quarto book?

We could move the intro content here and remove intro chapter?

# 1 Introduction

This book details all things having to do with code within DaSL. It includes both internal facing guidelines as well as for any contributions from folks other than DaSL staff.

- Getting help: [Chapter 2](#)
- Contributing guidelines: [Chapter 3](#)
- Style guidelines: [Chapter 4](#)
- Code review guidelines: [Chapter 5](#)
- Package documentation guidelines: [Chapter 6](#)

## 2 Help

Getting help ....

### 2.1 House Calls

XXXX

### 2.2 Slack

XXXX

## 3 Contributing

Contributing guidelines ....

### 3.1 Code of Conduct

xxxx

### 3.2 Communication Channels

- Slack

### 3.3 xxx

xxxx

## 4 Style

What is code style? It has to do with how you organize your code. The end goal of styling your code is that your code is more consistent. Styling code by definition has to be opinionated. You may not agree with every decision made in a style guide, but having style be done automatically for you gives you more time to think about the important decisions in your code.

If you always write code by yourself and you're the only one that will ever look at it, that's one thing. But that's rarely the case, especially if you consider that yourself 6 months or 5 years from now is a user that is keenly interested in readable, consistent code. Readable, consistent code will be especially appreciated by other users and contributors.

### 4.1 Package styler

The [styler](#) package allows you to interactively format your code according to the [tidyverse style guide](#).

The [lintr](#) package does automated checks of your code according to the style guide.

### 4.2 IDE and Text editor support

RStudio supports styler via the Addins drop down; see the [RStudio User Guide](#).

Support for `styler` in other editors is provided via the [R languageserver](#):

- VSCode: [vscode-R](#)
- Atom: [atom-ide-r](#)
- Sublime Text: [R-IDE](#)
- Vim/NeoVim: [LanguageClient-neovim](#)

See the [R languageserver](#) GitHub repository for more information on using the R language-server.

## 5 Code review

Code review ....

### 5.1 DaSL interal guidelines

XXXX

### 5.2 Community guidelines

XXXX



## 6 Package documentation

All formally supported DaSL R and Python packages should have package documentation.

### 6.1 R

We use the package [pkgdown](#) to create documentation for R packages.

For R packages, docs should be hosted on GitHub pages.

#### 6.1.1 DaSL pkgdown template

We are planning to have a DaSL specific `pkgdown` “package template” (see [pkgdown docs](#) for what this means) - but it’s not ready to use yet. When it is ready, you will be able to specify our template like:

```
template:  
  package: dasltemplate
```

For now just use the default theme that `pkgdown` provides.

### 6.2 Python

We use the package [Sphinx](#) to create documentation for Python packages.

For Python packages, docs should be hosted on [ReadTheDocs](#).

Sphinx is less automatic relative to `pkgdown`, so just be prepared for more manual work with Sphinx.

## 6.3 Guidelines

- README: This is most likely the first place potential users will interact with your package. Make sure the README clearly states what the package does, and how to get started.
- Examples: All user facing functions should have examples. Make sure to be careful about how examples are run if there's any sensitive data or connections to remote services.
- (anything else?)

## References