

# **RTI Connex DDS**

**Core Libraries**

**Getting Started Guide**

**Addendum for iOS Systems**

**Version 5.2.3**



© 2016 Real-Time Innovations, Inc.  
All rights reserved.  
Printed in U.S.A. First printing.  
April 2016.

## **Trademarks**

Real-Time Innovations, RTI, NDDS, RTI Data Distribution Service, DataBus, Connex, Micro DDS, the RTI logo, 1RTI and the phrase, “Your Systems. Working as one,” are registered trademarks, trademarks or service marks of Real-Time Innovations, Inc. All other trademarks belong to their respective owners.

## **Copy and Use Restrictions**

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form (including electronic, mechanical, photocopy, and facsimile) without the prior written permission of Real-Time Innovations, Inc. The software described in this document is furnished under and subject to the RTI software license agreement. The software may be used or copied only under the terms of the license agreement.

## **Technical Support**

Real-Time Innovations, Inc.

232 E. Java Drive

Sunnyvale, CA 94089

Phone: (408) 990-7444

Email: [support@rti.com](mailto:support@rti.com)

Website: <https://support.rti.com/>

---

<b>Chapter 1 Installing Connex DDS and Xcode .....</b>	<b>1</b>
<b>Chapter 2 Creating an Xcode Project .....</b>	<b>2</b>
<b>Chapter 3 Generating Example Code and an Xcode Project with rtiddsgen</b>	
3.1 About the Generated Code .....	5
<b>Chapter 4 Executing the Code .....</b>	<b>7</b>

# Chapter 1 Installing Connex DDS and Xcode

This document supplements the [Getting Started Guide](#) with additional steps for working with iOS® platforms.

- To install the Xcode® development software:

Download the software from the Apple® App Store® or developer website and follow the instructions.

- To install Connex DDS:

Follow the installation instructions in the [Getting Started Guide](#). Install the desired iOS architecture package(s).

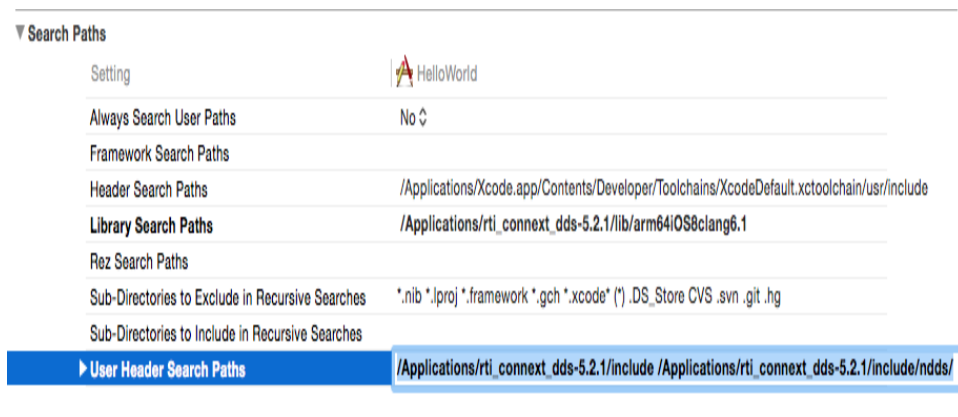
# Chapter 2 Creating an Xcode Project

1. Create a new iOS Project of whatever type is appropriate for your use case.

Follow instructions on the [Apple Developer's website \(https://developer-apple.com/library/ios/recipes/xcode\\_help-structure\\_navigator/articles/Creating\\_a\\_Project.html#//apple\\_ref/doc/uid/TP40009934-CH3-SW1\)](https://developer.apple.com/library/ios/recipes/xcode_help-structure_navigator/articles/Creating_a_Project.html#//apple_ref/doc/uid/TP40009934-CH3-SW1)

2. Add the Connex DDS core to your project:
  - a. Select the project and go to the **Build Settings** tab.
  - b. Add the path to the Connex DDS include files.

```
/Applications/rti_connex_dds-x.y.z/include /Applications/rti_connex_dds-x.y.z/include/ndds
```



- c. Add preprocessor definitions

Add the **RTI\_UNIX** preprocessor declaration and compiler option **-Wno-return-type-c-linkage**.

## ▼ Apple LLVM 6.0 - Custom Compiler Flags

Setting

| idlFile\_publisher

Other C Flags

-DRTI\_UNIX

Other C++ Flags

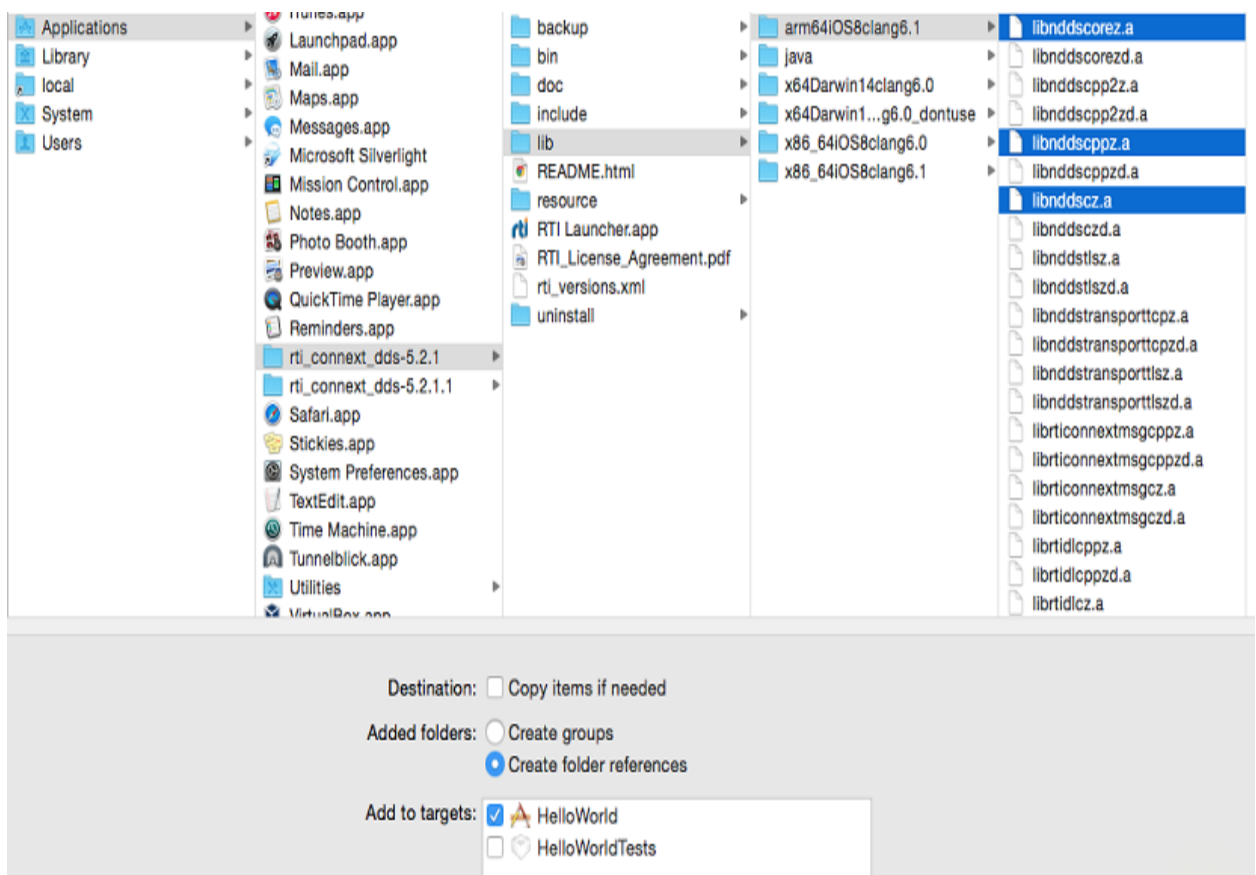
-DRTI\_UNIX

Other Warning Flags

-Wno-return-type-c-linkage

## d. Add the Connex DDS Libraries

To add the Connex DDS libraries, right-click on the project and select **Add Files to “<project name>”** or add them in the ‘Link Binary With Libraries’ section of the ‘Build Phases’ tab and choose **Add Other....** Either method will lead to a selection dialog like this:



Select the Connex DDS libraries for your architecture and language. All languages require **libnddscore** and **libnddsc**. For C++, add **libnddscpp**; etc. (see the iOS section of the *Platform Notes*). If you want to use any of the C++ APIs, rename the Xcode-generated **ViewController.m** source file (or whatever source file you use to call Connex DDS) to **ViewController.mm**.

### 3. Call the Connex DDS API

Call the Connex DDS *Entity* creation APIs (**create\_participant()**, **create\_topic()**, etc.) as described in the [Getting Started Guide](#). In the appropriate locations in your code, call the read and write operations.

### 4. Access the QoS files

Connex DDS locates the Quality of Service (QoS) XML file using a number of search paths (see the [User's Manual](#)). The Xcode development software transfers files to the target device via a settings bundle.

- a. Create a **Settings.bundle** resource following Apple's instructions, then place the QoS file(s) in the **Settings.bundle** folder.
- b. In your application code, do one of the following:
  - Set `NDDS_QOS_PROFILES` to the QoS file in the application's resource path.
  - Change the default directory to the application's resource bundle path.
  - Copy the QoS file(s) to the application's **documents** directory and change the default directory to the application's **documents** directory.

# Chapter 3 Generating Example Code and an Xcode Project with *rtiddsgen*

## 1. Run *rtiddsgen*:

From a terminal or ssh window, run *rtiddsgen* as per other Connex DDS architectures specifying the iOS architecture name.

## 2. Open the generated workspace or project:

*rtiddsgen* creates a workspace with two project files, one for the publisher and one for the subscriber. Open the workspace or project with the Xcode development software just as you would open any other Xcode project.

## 3.1 About the Generated Code

The code generated by *rtiddsgen* for iOS is slightly different than the code generated for other architectures.

- For most architectures, the main function, looping, and messaging are controlled completely by the application. However for an iOS architecture, the main event loop is within the operating environment framework.
- For an iOS architecture, the default output from **print()** goes to the debugger window.

It is important not to do any lengthy processing in the main loop of an iOS application. For this reason, the generated publisher code writes on a timer rather than in a 'for' loop with a sleep. Since the generated subscriber code uses waitsets and data is received in a middleware thread, the changes to the subscriber code are minimal. The generated subscriber project contains commented-out code that can redirect Connex DDS core messages to a text window. To enable this code, define REDIRECT\_LOGGING before building the application.



The generated examples place a default QoS file in the **Settings.bundle** folder and set the application's default directory to the resource path. (See [Access the QoS files \(Section on page 4\)](#)).

# Chapter 4 Executing the Code

Execute the application via the Xcode development software, just as you would execute any other iOS application.