



# Bolt Studio Abaqus/CAE Plug-in

---

## Introduction

The Bolt Studio plug-in provides the user with a streamlined method for defining bolts, nuts and washers, and placing them into an existing Abaqus/CAE model. Users can control the default set of bolts that are displayed in the interface via a simple Python-based configuration file. The bolts, nuts and washers, where applicable, are generated parametrically within Abaqus/CAE, and meshed automatically using a hexahedral mesh. The mesh size is determined automatically based on the dimensions of the bolt (the user can re-mesh, if required, the parts using the native meshing tools within Abaqus/CAE). The bolt is automatically partitioned and the specified pre-loading applied.

## Installation

To install the Bolt Studio plug-in, unzip the file **boltStudio.zip** into any of the Abaqus/CAE accessible `abaqus_plugins` directories. The files can be added to any of the following locations:

- `abaqus_dir\abaqus_plugins`, where *abaqus\_dir* is the Abaqus parent directory.
- `abaqus_dir\code\python\lib\abaqus_plugins`, where *abaqus\_dir* is the Abaqus parent directory.
- `home_dir\abaqus_plugins`, where *home\_dir* is your home directory.
- `current_dir\abaqus_plugins`, where *current\_dir* is the current directory.
- `plugin_dir`, where *plugin\_dir* is a directory specified in the `abaqus_v6.env` file by the environment variable **plugin\_central\_dir**. You can store plug-ins in a central location that can be accessed by all users at your site if the directory to which **plugin\_central\_dir** refers is mounted on a file system that all users can access. For example,

```
plugin_central_dir = r'\\fileServer\sharedDirectory'
```

For more information, see “Using the Abaqus environment file,” Section 4.1 of the Abaqus Installation and Licensing Guide.

## Controlling the list of bolts shown in Bolt Studio

The file named `boltStudioDefaults.py` in the `boltStudio` directory is used to control the default bolts that are displayed in the ‘Bolt Family’ and ‘Bolt Type’ drop-down menus. The data in the file is defined as a Python dictionary. An example of the file structure can be seen in Figure 1:

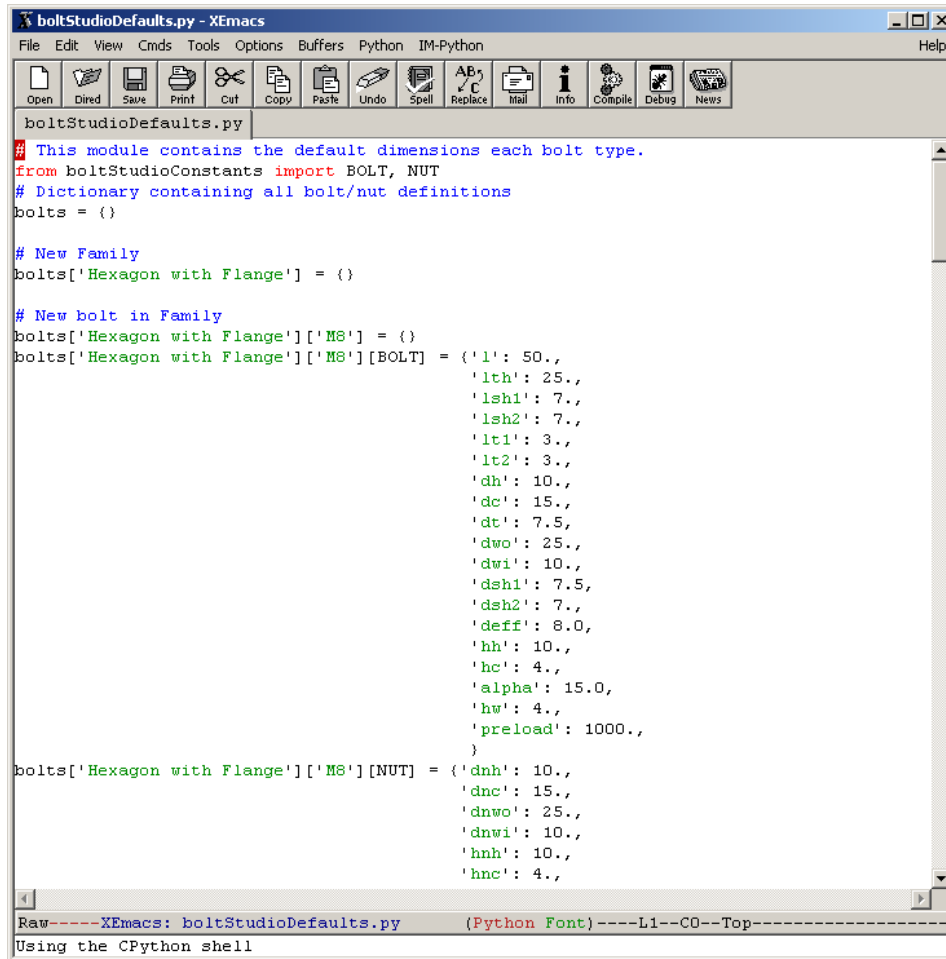


Figure 1: Default file structure, boltStudioDefaults.py

The ‘bolts’ dictionary contains all of the parametric information. The first key in the dictionary identifies the bolt family into which the bolt being defined will be placed. This key will be displayed in the ‘Bolt Family’ combo box. The second key specifies the bolt name. This key will be displayed in the ‘Bolt Type’ combo box. The third key is defined as either BOLT or NUT and specifies the sub-dictionary that contains information for the bolt and nut respectively. The user can add as many bolt families and bolt types as they wish. However, the dictionary structure must adhere to the rules shown above, and must be a valid dictionary.

## Using Bolt Studio

The Bolt Studio plug-in can be started in any Abaqus/CAE module that displays the model assembly in the viewport. To start the plug-in, press **Plug-ins → Bolt Studio....** The Bolt Studio dialog box, shown in Figure 2, will be displayed.

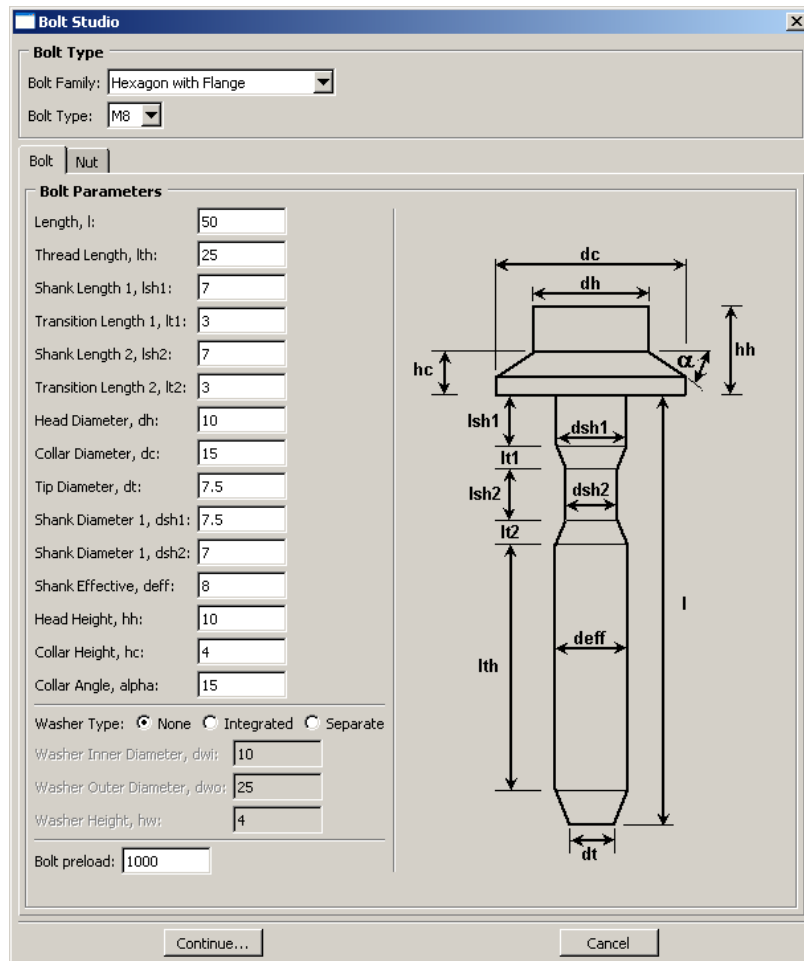


Figure 2: Bolt Studio dialog

The user can control the bolt family and bolt type to use by using the combo boxes at the top of the dialog. When a bolt type is selected from the combo box, the dialog is automatically updated to use the values specified for that bolt type. The main portion of the dialog box contains two tabs: one for the bolt definition and one for the nut definition. The user can override the values from the selected bolt type (for example, changing the bolt length) but selecting the parameter text box and changing the value. When the user selects an item to modify, the graphic is updated, as shown in Figure 3, to indicate clearly the parameter that is being modified.

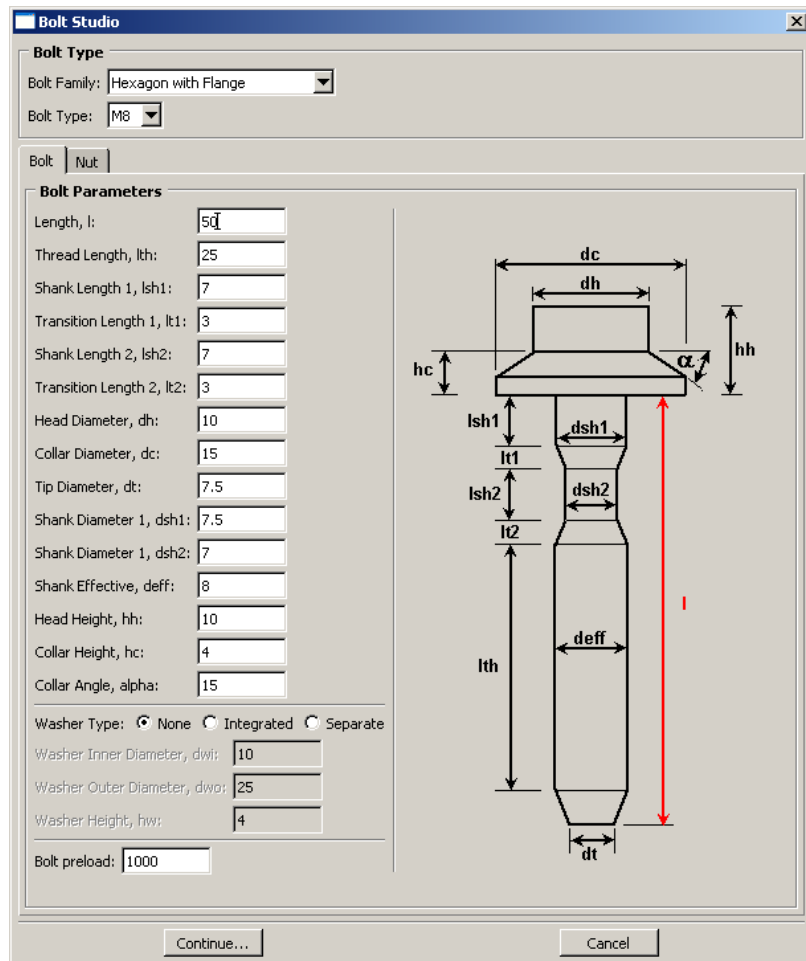


Figure 3: Changing a parameter

The user can also choose whether or not to include a washer in the assembly. Two washer options are given: integrated and separate. The bolt diagram will be updated according to the selection, as seen in Figure 4 for the integrated and separated types. If an integrated washer is chosen, the bolt and washer will be generated as a single component. If a separate washer is chosen, individual parts will be generated for the bolt and washer.

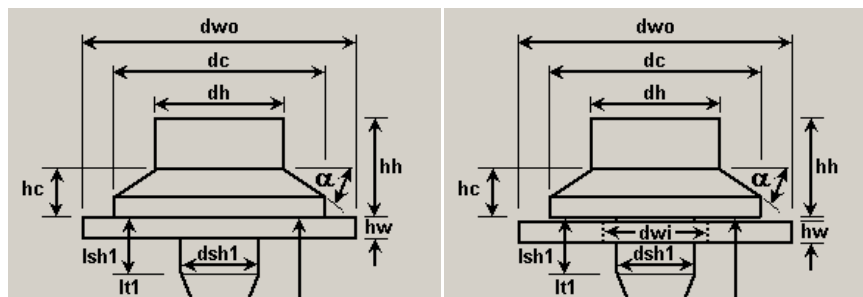


Figure 4: Washer types

From the bolt tab the user can also specify the pre-load to be applied to the bolt. When the bolt is placed in the assembly, the bolt pre-load is automatically defined. If there are existing steps in the database, the pre-loading is applied in the first step, and the bolt length fixed in the second step. If no steps exist in the model, they will be generated automatically by the plug-in.

Once the bolt dimensions have been defined, the user can switch to the 'Nut' tab where they can control the nut definition (if applicable). The nut tab is shown in . The nut tab behaves similarly to the bolt tab with a two minor differences. These are i) the user can choose to include the nut or not and ii) the user can copy the nut and washer definition parameters from those defined in the bolt tab.

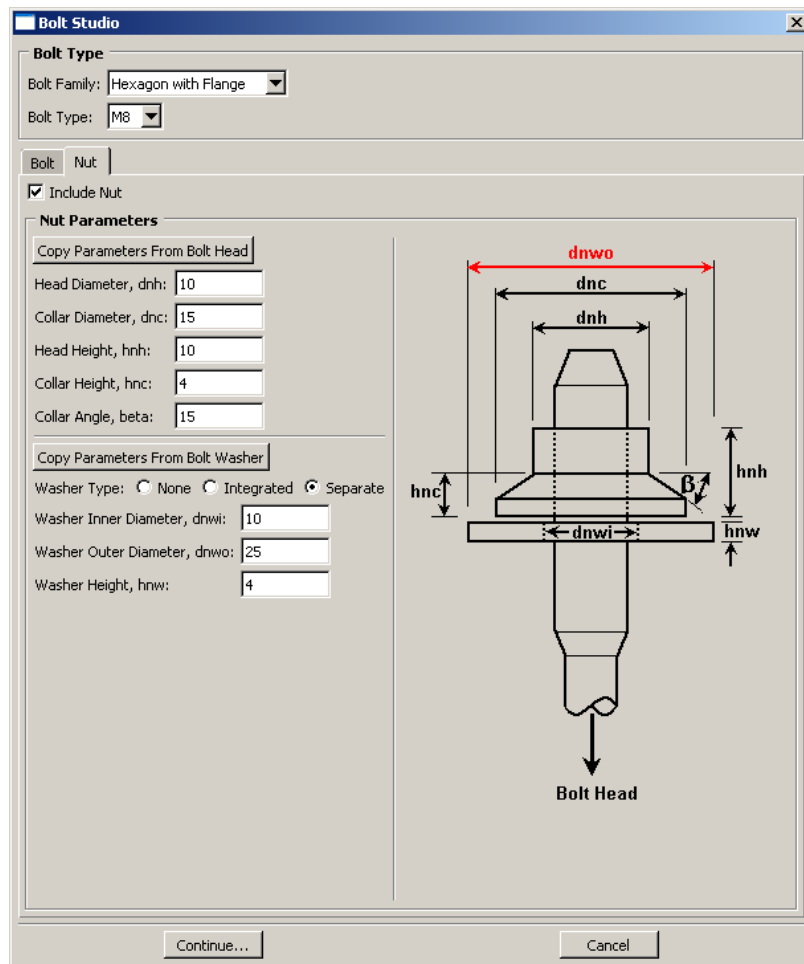


Figure 5: Nut definition tab

Once the user has defined the bolt and nut definitions, the 'Continue...' button is pressed to start the process of positioning the bolt/stud and, where applicable nuts and separate washers into the assembly. Once the 'Continue...' button has been pressed, the dialog box pops-down and the user is led through the placement process via a series of questions in the prompt area of the Abaqus/CAE main window. Once this process is complete, the bolt/stud, nut and washers are created and positioned in the assembly (the bolt loads are also applied). The series of questions is then repeated from the beginning in order that multiple bolts of the same type

can be placed in the assembly. Once the user has placed all of the bolts in the assembly, they can exit the placement process by pressing the cancel button in the prompt area.

The series of placement questions are dynamically modified based on the users answers to previous questions. For example, to specify the axis to which the bolt axis will be aligned, the user is asked to “Select bolt axis, face for concentric constraint, or edge on top of cylinder”. If the user selects the third choice, they would then be prompted to “Select edge on bottom of cylinder” from which the axis would be calculated. In cases where the user is asked to select a face or edge, they can select either geometric faces/edges or element faces/edges. If the user specifies a nut to be defined, an additional question regarding the positioning of the nut face is also asked. A flowchart showing the series of questions is shown in Figure 6.

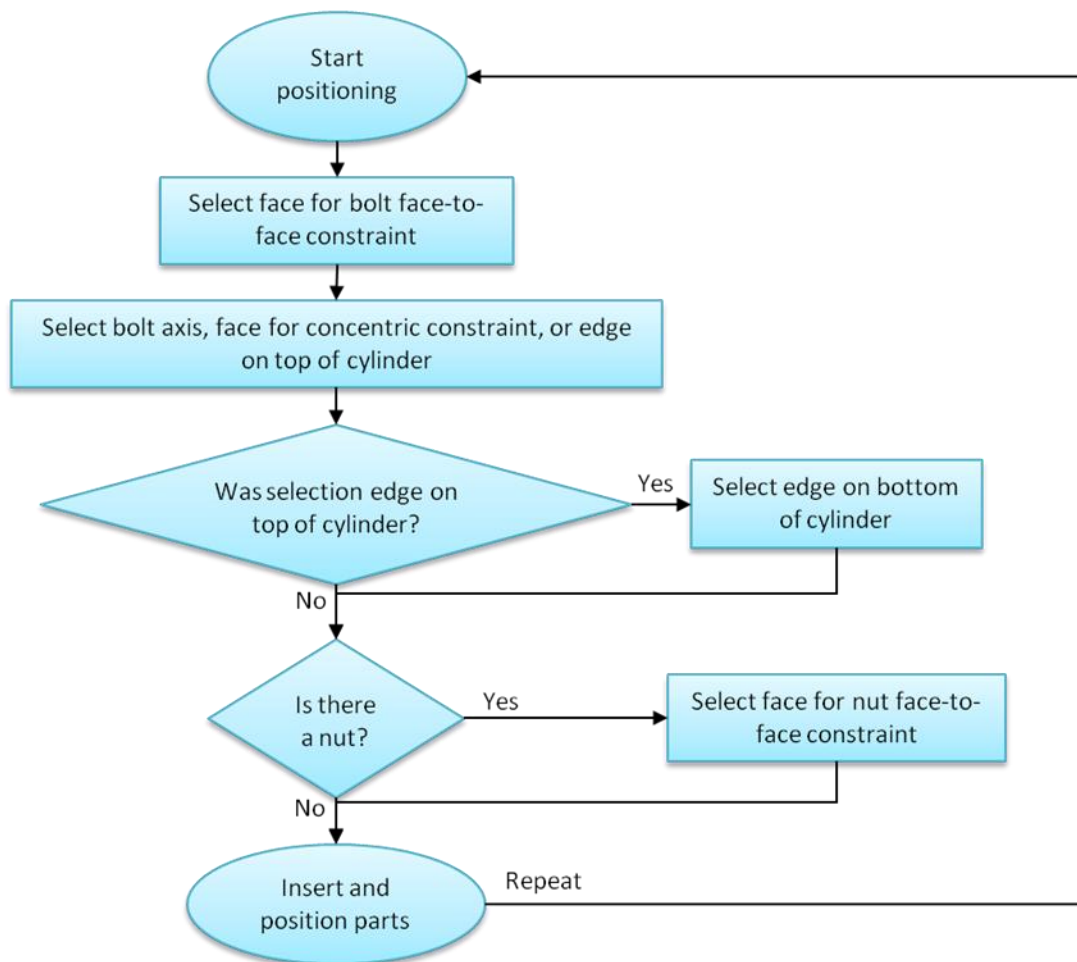
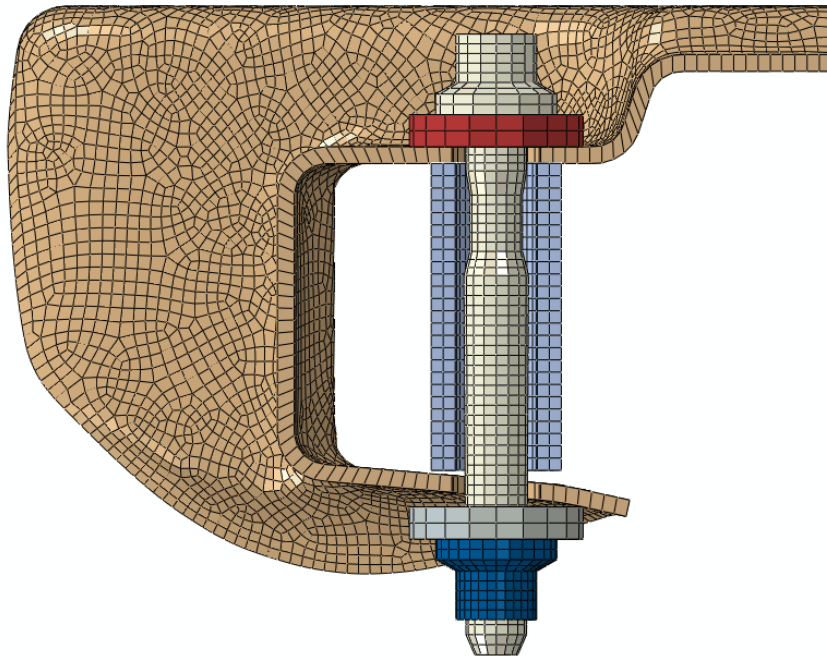


Figure 6: Positioning flowchart

Figure shows an example of a nut, bolt and separate washers added to an assembly (half of which is removed for display purposes) using the plug-in. The intention of this analysis is to perform a closing of the bracket from its initially splayed configuration. The plug-in positions the nut washer such that the washer is just touching the oblique face of the bracket.



*Figure 7: Bolt, nut and washers in assembly*