FreeCAD Fasteners Python API Guide

Summary of Common Errors and Solutions

This guide documents the correct way to create fasteners programmatically in FreeCAD, based on troubleshooting sessions.

❌ Common Mistakes to Avoid

1. Wrong Module Name

python

```python
# ❌ WRONG - Module is not called 'fasteners'
import fasteners

# ✅ CORRECT - Use FastenersCmd (capital letters)
import FastenersCmd
```

2. Wrong Function Names

python

```python
# ❌ WRONG - These functions don't exist
FastenersCmd.FSCreateHexHeadScrew('ISO4017', 'M8', '40')
FreeCADGui.runCommand('FSHexScrew')

# ✅ CORRECT - Use the object constructors
FastenersCmd.FSScrewObject(obj, "ISO4017", None)
```

3. Wrong Parameter Order

python

```python
# ❌ WRONG - Passing strings directly
screw = FastenersCmd.FSScrewObject("ISO4017", "M8-40", None)

# ✅ CORRECT - First parameter must be a FreeCAD object
obj = doc.addObject("Part::FeaturePython", "Screw")
FastenersCmd.FSScrewObject(obj, "ISO4017", None)
```

4. Missing View Provider

```python
# ❌ WRONG - Fastener won't be visible in 3D view
obj = doc.addObject("Part::FeaturePython", "Screw")
FastenersCmd.FSScrewObject(obj, "ISO4017", None)

# ✅ CORRECT - Add view provider for visibility
obj = doc.addObject("Part::FeaturePython", "Screw")
FastenersCmd.FSScrewObject(obj, "ISO4017", None)
FastenersCmd.FSViewProviderTree(obj.ViewObject)
```

5. No Active Document

```python
# ❌ WRONG - Assuming document exists
doc = FreeCAD.ActiveDocument
obj = doc.addObject(...)  # Crashes if doc is None

# ✅ CORRECT - Check and create if needed
doc = FreeCAD.ActiveDocument
if not doc:
    doc = FreeCAD.newDocument("MyDocument")
```

✅ Correct Pattern for Creating Fasteners

Basic Template

```python
```

```
import FreeCAD
import FreeCADGui
import FastenersCmd

# 1. Get or create document
doc = FreeCAD.ActiveDocument
if not doc:
    doc = FreeCAD.newDocument("FastenerProject")

# 2. Create FreeCAD object
obj = doc.addObject("Part::FeaturePython", "MyFastener")

# 3. Initialize fastener (choose one based on type)
FastenersCmd.FSScrewObject(obj, "ISO4017", None)  # For screws/bolts/nuts
# OR
FastenersCmd.FSWasherObject(obj, "ISO7089", None)  # For washers

# 4. Add view provider for visualization
FastenersCmd.FSViewProviderTree(obj.ViewObject)

# 5. Recompute and display
doc.recompute()
FreeCADGui.SendMsgToActiveView("ViewFit")
```

📋 Available Fastener Classes

Based on the FastenersCmd module, here are the main object types:

Class	Purpose	Example
`FSScrewObject`	Screws, bolts, and nuts	ISO4017, ISO4032
`FSWasherObject`	Washers	ISO7089, ISO7093
`FSScrewRodObject`	Threaded rods	DIN975, DIN976
`FSThreadedRodObject`	Threaded rods	Various standards
`FSScrewDieObject`	Dies	Not commonly used

🔧 Working with Fastener Properties

Finding Available Properties

After creating a fastener, inspect its properties:

```python
obj = doc.addObject("Part::FeaturePython", "Screw")
FastenersCmd.FSScrewObject(obj, "ISO4017", None)

# List all properties
print("Available properties:")
for prop in obj.PropertiesList:
    try:
        value = getattr(obj, prop)
        print(f"  {prop}: {value}")
    except:
        print(f"  {prop}: <cannot read>")
```

Common Properties (varies by fastener type)

- `diameter` - Thread diameter (e.g., "M8", "M10")
- `length` - Fastener length (e.g., "40 mm", "50 mm")
- `type` - Fastener standard (e.g., "ISO4017")
- `invert` - Boolean to flip orientation
- `offset` - Offset value

Note: Property names may vary. Always inspect first!

📦 Complete Examples

Example 1: Single Hex Bolt

```python
```

```python
import FreeCAD
import FreeCADGui
import FastenersCmd

doc = FreeCAD.ActiveDocument
if not doc:
    doc = FreeCAD.newDocument("BoltTest")

obj = doc.addObject("Part::FeaturePython", "HexBolt")
FastenersCmd.FSScrewObject(obj, "ISO4017", None)
FastenersCmd.FSViewProviderTree(obj.ViewObject)

doc.recompute()
FreeCADGui.SendMsgToActiveView("ViewFit")
print("Hex bolt created!")
```

Example 2: Multiple Fasteners (Bolt + Washer + Nut)

python

```
import FreeCAD
import FreeCADGui
import FastenersCmd

doc = FreeCAD.ActiveDocument
if not doc:
    doc = FreeCAD.newDocument("Assembly")

# Create hex bolt
bolt = doc.addObject("Part::FeaturePython", "Bolt")
FastenersCmd.FSScrewObject(bolt, "ISO4017", None)
FastenersCmd.FSViewProviderTree(bolt.ViewObject)
bolt.Label = "M8x40 Hex Bolt"

# Create washer
washer = doc.addObject("Part::FeaturePython", "Washer")
FastenersCmd.FSWasherObject(washer, "ISO7089", None)
FastenersCmd.FSViewProviderTree(washer.ViewObject)
washer.Label = "M8 Washer"

# Create nut
nut = doc.addObject("Part::FeaturePython", "Nut")
FastenersCmd.FSScrewObject(nut, "ISO4032", None)
FastenersCmd.FSViewProviderTree(nut.ViewObject)
nut.Label = "M8 Hex Nut"

doc.recompute()
FreeCADGui.SendMsgToActiveView("ViewFit")
print("Assembly created!")
```

Example 3: Finding Existing Objects

python

```
import FreeCAD

doc = FreeCAD.ActiveDocument
if doc:
    print("Objects in document:")
    for obj in doc.Objects:
        print(f"  Name: {obj.Name}, Label: {obj.Label}, Type: {obj.TypeId}")

    # Find specific object
    screw_obj = None
    for obj in doc.Objects:
        if "Screw" in obj.Label:
            screw_obj = obj
            break

    if screw_obj:
        print(f"\nFound screw: {screw_obj.Label}")
else:
    print("No active document")
```

🔍 Useful Fastener Groups and Standards

The module organizes fasteners into groups:

- `HexHeadGroup` - Hex head bolts
- `HexagonSocketGroup` - Socket head cap screws
- `NutGroup` - Various nuts
- `WasherGroup` - Washers
- `ThreadedRodGroup` - Threaded rods
- `SetScrewGroup` - Set screws
- `SlottedGroup` - Slotted screws
- `PinGroup` - Pins
- `RetainingRingGroup` - Retaining rings

Common Standards:

- **ISO 4017** - Hex head bolts

- **ISO 4032** - Hex nuts
- **ISO 7089** - Washers
- **DIN 912** - Socket head cap screws
- **DIN 934** - Hex nuts

🛠️ Debugging Tips

1. **Check if module is loaded:**

```python
import FastenersCmd
print(dir(FastenersCmd))
```

2. **Verify workbench is available:**

```python
import FreeCADGui
print('FastenersWorkbench' in FreeCADGui.listWorkbenches())
```

3. **Always recompute after changes:**

```python
doc.recompute()
```

4. **Use try-except for robust code:**

```python
try:
    obj = doc.addObject("Part::FeaturePython", "Fastener")
    FastenersCmd.FSScrewObject(obj, "ISO4017", None)
    FastenersCmd.FSViewProviderTree(obj.ViewObject)
    doc.recompute()
except Exception as e:
    print(f"Error creating fastener: {e}")
```

📝 Key Takeaways

1. **Always use `FastenersCmd`** (not `fasteners`)
2. **Create FreeCAD object first**, then initialize fastener class
3. **Add `FSViewProviderTree`** for 3D visualization
4. **Check for active document** before creating objects
5. **Use `doc.recompute()`** after creating/modifying fasteners
6. **Inspect properties** before trying to set them
7. **Third parameter is `attachTo`** - use `None` for standalone fasteners

---

🚀 Where to Use This Code

- **Python Console** - For single-line commands only
- **Macro Editor** - Best for multi-line scripts (Macro → Macros → Create)
- **External .py files** - Execute via Macro → Execute macro

Last Updated: Based on FreeCAD 1.0.2 with Fasteners Workbench