Working with Meshes

Table of contents

A Mesh record defines the Coreform IGA discretization for one Abaqus part or independent assembly instance. Each valid owner stores at most one persistent Mesh.

1 Choose the Mesh owner

A part and all its assembly instances form one ownership family. Store the Mesh according to that family's dependency state:

Part-family state	Mesh owner
The part has no instances	The part
All instances are dependent	The part; its Mesh governs the dependent instances
All instances are independent	Each independent instance that needs a Mesh

Part-owned and instance-owned Meshes cannot coexist in one family. A dependent instance cannot own a Mesh directly, and an independent instance cannot use a Mesh stored on its part.

2 Choose the Mesh type

An `ImmersedRectilinearMesh` creates a rectilinear background discretization trimmed to the owner's geometry. Choose exactly one sizing strategy: element size, total interval counts, or per-axis interval counts. You can provide a complete coordinate frame or let Coreform derive one from the owner.

A `BodyFitMesh` records a body-fitting intent and does not accept the rectilinear sizing, layout, padding, or frame controls.

3 Create and resolve an immersed Mesh

```
from abaqus import mdb
import coreform.abaqus

coreform.abaqus.init_iga_mdb()

model = mdb.models["Beam"]
part = model.parts["Beam"]
meshes = part.customData.CoreformIGA

mesh = meshes.ImmersedRectilinearMesh(
    degree=2,
    continuity=1,
    elementSize=(0.5, 0.5, 0.5),
    elementLayout=("node_centered",) * 3,
    smallCellVolumeRatio=0.2,
)

mesh.setValues(
    degree=3,
    continuity=2,
    elementSize=(0.25, 0.25, 0.25),
)

resolved = mesh.resolveForOwner()
print(resolved.frame.origin)
print(resolved.frame.basis)
print(resolved.extents)
print(resolved.padding)
```

`toSpec()` preserves automatic frame and padding fields as `None`. `resolveForOwner()` returns a portable specification with owner-dependent values materialized without changing the stored Mesh.

4 Replace or remove a Mesh

Creating a second Mesh on the same owner normally raises `ValidationError`. Use `overwrite=True` when replacement is intentional; the candidate is fully validated before the existing record changes.

```
replacement = meshes.BodyFitMesh(
    degree=2,
    continuity=1,
```

```
    overwrite=True,
)

meshes.deleteMesh()
```

See the Abaqus Mesh reference and portable Mesh reference for exact constructors, sizing and frame groups, record attributes, and namespace operations.