coreform.abaqus.probe

Table of contents

Source directory: `interop/abaqus/cf_app/coreform/abaqus/probe`

Directory tree

- `probe/`

1 records.py

Module path: `coreform.abaqus.probe.records`

Available API members

Name	Kind	Description
`CoreformIGA.probes`	Attribute	Persistent Probe records indexed by name.
`probes[probeName].name`	Attribute	Repository key and Probe name.
`probes[probeName].probeType`	Attribute	POINT_PROBE, LINE_PROBE, or MULTI_POINT_PROBE.

Name	Kind	Description
probes[probeName].instanceNames	Attribute	Selected instance names, or None when another target kind is active.
probes[probeName].setNames	Attribute	Selected assembly-set names, or None when another target kind is active.
probes[probeName].partNames	Attribute	Selected part names, or None when another target kind is active.
probes[probeName].variables	Attribute	Selected output variables, or None when the sequence is empty.
probes[probeName].instances	Attribute	Resolved Abaqus instances, or an empty tuple when not targeted.
probes[probeName].sets	Attribute	Resolved Abaqus assembly sets, or an empty tuple when not targeted.
probes[probeName].parts	Attribute	Resolved Abaqus parts, or an empty tuple when not targeted.
probes[probeName].location	Attribute	Point at which the Probe evaluates its variables.
probes[probeName].startLocation	Attribute	First endpoint of the Probe line segment.
probes[probeName].stopLocation	Attribute	Second endpoint of the Probe line segment.
probes[probeName].numEvalPoints	Attribute	Number of evaluation points along the line, including both endpoints.
probes[probeName].locations	Attribute	Explicit locations at which the Probe evaluates its variables.

Name	Kind	Description
`CoreformIGA.model`	Instance method	Return the Abaqus Model that owns this Probe namespace.
`CoreformIGA.PointProbe`	Instance method	Create and persist a point Probe in this Steps namespace.
`CoreformIGA.LineProbe`	Instance method	Create and persist a line Probe in this Steps namespace.
`CoreformIGA.MultiPointProbe`	Instance method	Create and persist a multi-point Probe in this Steps namespace.
`CoreformIGA.deleteProbe`	Instance method	Delete a named Probe from this namespace.
`probes[probeName].model`	Instance method	Return the Abaqus Model that owns this Probe record.
`probes[probeName].toSpec`	Instance method	Return the immutable portable specification for this Probe.
`probes[probeName].validate`	Instance method	Validate the Probe's target names against its owning model.
`probes[probeName].setValues`	Instance method	Update this persistent Probe in place.
`CoreformIGA`	Class	Persistent Probe namespace attached to an Abaqus model's Steps repository.
`probes[probeName]`	Class	Persistent probe record with loaded public fields treated as read-only.

1.1 `CoreformIGA`

Public path: `mdb.models[modelName].steps.customData.CoreformIGA`

Available API members

Name	Kind	Description
`probes`	Attribute	Persistent Probe records indexed by name.
`model`	Instance method	Return the Abaqus Model that owns this Probe namespace.
`PointProbe`	Instance method	Create and persist a point Probe in this Steps namespace.
`LineProbe`	Instance method	Create and persist a line Probe in this Steps namespace.
`MultiPointProbe`	Instance method	Create and persist a multi-point Probe in this Steps namespace.
`deleteProbe`	Instance method	Delete a named Probe from this namespace.

1.1.a Attributes

Name	Type	Value	Description
`probes`	`object`	—	Persistent Probe records indexed by name.

1.1.b `model`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.model`

Description: Return the Abaqus Model that owns this Probe namespace.

Call syntax

```
result = probes.model()
```

Returns

Type	Description
`object`	Return the Abaqus Model that owns this Probe namespace.

Signatures

```
model() -> object
```

Parameters

None.

1.1.c `PointProbe`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.PointProbe`

Description: Create and persist a point Probe in this Steps namespace.

Call syntax

```
result = probes.PointProbe(spec=spec)
result = probes.PointProbe(name=name, instanceNames=instanceNames,
location=location)
```

Returns

Type	Description
`PointProbe record`	Newly created persistent point-Probe record.

Signatures

```
PointProbe(*, spec) -> object
PointProbe(*, name, instanceNames=None, setNames=None, partNames=None,
variables=(), location) -> object
```

Parameters

For `PointProbe(*, spec) -> object`:

Name	Type	Required	Default	Description
`spec`	`PointProbeSpec` or `Mapping[str, object]`	Yes	—	Complete portable specification or serialized specification mapping.

For `PointProbe(*, name, instanceNames=None, setNames=None, partNames=None, variables=(), location) -> object`:

Name	Type	Required	Default	Description
`name`	`str`	Yes	—	Unique, non-empty name in the Probe repository.
`instanceNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Assembly-instance targets.
`setNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Assembly-set targets.
`partNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Model-part targets.
`variables`	`Sequence[str]`	No	`()`	Output-variable names to evaluate. The default is an empty sequence.
`location`	`Point3D`	Yes	—	Evaluation location.

Parameter rules

Rule	Requirement
Call forms	Use exactly one of these call forms: `spec` or `abaqus_definition`.
Required group	Supply exactly one of `instanceNames`, `setNames`, or `partNames`.

Raises

Type	Condition
`ValidationError`	If the specification, target selection, name, or location is invalid.

Example

```
>>> probe = probes.PointProbe(
...     name="Probe-1",
...     instanceNames=("PART-1-1",),
...     location=(0.0, 0.0, 0.0),
... )
```

1.1.d `LineProbe`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.LineProbe`

Description: Create and persist a line Probe in this Steps namespace.

Call syntax

```
result = probes.LineProbe(spec=spec)
result     =     probes.LineProbe(name=name,     instanceNames=instanceNames,
startLocation=startLocation, stopLocation=stopLocation)
```

Returns

Type	Description
`LineProbe record`	Newly created persistent line-Probe record.

Signatures

```
LineProbe(*, spec) -> object
LineProbe(*,   name,   instanceNames=None,   setNames=None,   partNames=None,
variables=(), startLocation, stopLocation, numEvalPoints=2) -> object
```

Parameters

For `LineProbe(*, spec) -> object`:

Name	Type	Required	Default	Description
`spec`	`LineProbeSpec or Mapping[str, object]`	Yes	—	Complete portable specification or se-

Name	Type	Required	Default	Description
				rialized specification mapping.

For `LineProbe(*, name, instanceNames=None, setNames=None, partNames=None, variables=(), startLocation, stopLocation, numEvalPoints=2) -> object`:

Name	Type	Required	Default	Description
`name`	`str`	Yes	—	Unique, nonempty name in the Probe repository.
`instanceNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Assembly-instance targets.
`setNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Assembly-set targets.
`partNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Model-part targets.
`variables`	`Sequence[str]`	No	`()`	Output-variable names to evaluate. The default is an empty sequence.
`startLocation`	`Point3D`	Yes	—	First endpoint of the line segment.
`stopLocation`	`Point3D`	Yes	—	Second endpoint of the line segment; it must differ

Name	Type	Required	Default	Description
				from startLocation.
`numEvalPoints`	`int`	No	`2`	Number of evaluation points, including endpoints; it must be at least 2.

Parameter rules

Rule	Requirement
Call forms	Use exactly one of these call forms: `spec` or `abaqus_definition`.
Required group	Supply exactly one of `instanceNames`, `setNames`, or `partNames`.

Raises

Type	Condition
`ValidationError`	If the specification, target selection, name, or geometry is invalid.

1.1.e `MultiPointProbe`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.MultiPointProbe`

Description: Create and persist a multi-point Probe in this Steps namespace.

Call syntax

```
result = probes.MultiPointProbe(spec=spec)
result = probes.MultiPointProbe(name=name, instanceNames=instanceNames, locations=locations)
```

Returns

Type	Description
`MultiPointProbe record`	Newly created persistent multi-point-Probe record.

Signatures

```
MultiPointProbe(*, spec) -> object
MultiPointProbe(*, name, instanceNames=None, setNames=None, partNames=None,
variables=(), locations) -> object
```

Parameters

For `MultiPointProbe(*, spec) -> object`:

Name	Type	Required	Default	Description
`spec`	`MultiPointProbeSpec` or `Mapping[str, object]`	Yes	—	Complete portable specification or serialized specification mapping.

For `MultiPointProbe(*, name, instanceNames=None, setNames=None, partNames=None, variables=(), locations) -> object`:

Name	Type	Required	Default	Description
`name`	`str`	Yes	—	Unique, non-empty name in the Probe repository.
`instanceNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Assembly-instance targets.
`setNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Assembly-set targets.
`partNames`	`Sequence[str]`	Exactly one of `instanceNames`, `setNames`, or `partNames`	`None`	Model-part targets.
`variables`	`Sequence[str]`	No	`()`	Output-variable names

Name	Type	Required	Default	Description
				to evaluate. The default is an empty sequence.
`locations`	`Sequence[Point3D]`	Yes	—	One or more explicit evaluation locations.

Parameter rules

Rule	Requirement
Call forms	Use exactly one of these call forms: `spec` or `abaqus_definition`.
Required group	Supply exactly one of `instanceNames`, `setNames`, or `partNames`.

Raises

Type	Condition
`ValidationError`	If the specification, target selection, name, or locations are invalid.

1.1.f `deleteProbe`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.deleteProbe`

Description: Delete a named Probe from this namespace.

Call syntax

```
probes.deleteProbe(name=name)
```

Returns

Type	Description
`None`	—

Signatures

```
deleteProbe(name: str) -> None
```

Parameters

Name	Type	Required	Default	Description
`name`	`str`	Yes	—	Name of the Probe record to delete.

Raises

Type	Condition
`KeyError`	If the repository does not contain the supplied name.

1.2 `probes[probeName]`

Public path: `mdb.models[modelName].steps.customData.CoreformIGA.probes[probeName]`

Available API members

Name	Kind	Description
`name`	Attribute	Repository key and Probe name.
`probeType`	Attribute	POINT_PROBE, LINE_PROBE, or MULTI_POINT_PROBE.
`instanceNames`	Attribute	Selected instance names, or None when another target kind is active.
`setNames`	Attribute	Selected assembly-set names, or None when another target kind is active.
`partNames`	Attribute	Selected part names, or None when another target kind is active.
`variables`	Attribute	Selected output variables, or None when the sequence is empty.
`instances`	Attribute	Resolved Abaqus instances, or an empty tuple when not targeted.

Name	Kind	Description
`sets`	Attribute	Resolved Abaqus assembly sets, or an empty tuple when not targeted.
`parts`	Attribute	Resolved Abaqus parts, or an empty tuple when not targeted.
`location`	Attribute	Point at which the Probe evaluates its variables.
`startLocation`	Attribute	First endpoint of the Probe line segment.
`stopLocation`	Attribute	Second endpoint of the Probe line segment.
`numEvalPoints`	Attribute	Number of evaluation points along the line, including both endpoints.
`locations`	Attribute	Explicit locations at which the Probe evaluates its variables.
`model`	Instance method	Return the Abaqus Model that owns this Probe record.
`toSpec`	Instance method	Return the immutable portable specification for this Probe.
`validate`	Instance method	Validate the Probe's target names against its owning model.
`setValues`	Instance method	Update this persistent Probe in place.

1.2.a Attributes

Name	Type	Applies to	Value	Description
`name`	`str`	All	—	Repository key and Probe name.
`probeType`	`object`	All	—	POINT_PROBE, LINE_PROBE, or MULTI_POINT_PROBE.
`instanceNames`	`Optional[Tuple[str, ...]]`	All	—	Selected instance names, or None when another target kind is active.
`setNames`	`Optional[Tuple[str, ...]]`	All	—	Selected assembly-set names, or None when another target kind is active.
`partNames`	`Optional[Tuple[str, ...]]`	All	—	Selected part names, or None when another target kind is active.
`variables`	`Optional[Tuple[str, ...]]`	All	—	Selected output variables, or None when the sequence is empty.
`instances`	`tuple[object, ...]`	All	—	Resolved Abaqus instances, or an empty tuple when not targeted.
`sets`	`tuple[object, ...]`	All	—	Resolved Abaqus as-

Name	Type	Applies to	Value	Description
				sembly sets, or an empty tuple when not targeted.
`parts`	`tuple[object, ...]`	All	—	Resolved Abaqus parts, or an empty tuple when not targeted.
`location`	`Point3D`	`PointProbe`	—	Point at which the Probe evaluates its variables.
`startLocation`	`Point3D`	`LineProbe`	—	First endpoint of the Probe line segment.
`stopLocation`	`Point3D`	`LineProbe`	—	Second endpoint of the Probe line segment.
`numEvalPoints`	`int`	`LineProbe`	—	Number of evaluation points along the line, including both endpoints.
`locations`	`Tuple[Point3D, ...]`	`MultiPointProbe`	—	Explicit locations at which the Probe evaluates its variables.

1.2.b `model`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.probes[probeName].model`

Description: Return the Abaqus Model that owns this Probe record.

Call syntax

```
result = probe.model()
```

Returns

Type	Description
`object`	Return the Abaqus Model that owns this Probe record.

Signatures

```
model() -> object
```

Parameters

None.

1.2.c `toSpec`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.probes[probeName].toSpec`

Description: Return the immutable portable specification for this Probe.

Call syntax

```
result = probe.toSpec()
```

Returns

Type	Description
`ProbeSpec`	Return the immutable portable specification for this Probe.

Signatures

```
toSpec() -> ProbeSpec
```

Parameters

None.

1.2.d `validate`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.probes[probeName].validate`

Description: Validate the Probe's target names against its owning model.

Call syntax

```
probe.validate()
```

Returns

Type	Description
`None`	—

Signatures

```
validate() -> None
```

Parameters

None.

1.2.e `setValues`

Defined on: `mdb.models[modelName].steps.customData.CoreformIGA.probes[probeName].setValues`

Description: Update this persistent Probe in place.

Call syntax

```
probe.setValues(...)
```

Returns

Type	Description
`None`	The record is updated in place.

Signatures

```
PointProbe.setValues(*,     name=...,     instanceNames=...,     setNames=...,
partNames=..., variables=..., location=...) -> None
LineProbe.setValues(*,     name=...,     instanceNames=...,     setNames=...,
partNames=...,     variables=...,     startLocation=...,     stopLocation=...,
numEvalPoints=...) -> None
MultiPointProbe.setValues(*,    name=...,    instanceNames=...,    setNames=...,
partNames=..., variables=..., locations=...) -> None
```

Parameters

For `PointProbe.setValues(*, name=..., instanceNames=..., setNames=..., partNames=..., variables=..., location=...) -> None`:

Name	Applies to	Type	Required	Default	Description
`name`	All	`str or None`	No	Current value	May be omitted, empty, `None`, or unchanged; Probe renaming is not supported.
`instanceNames`	All	`Sequence[str]`	No	Current value	Switch the target to assembly instances.
`setNames`	All	`Sequence[str]`	No	Current value	Switch the target to assembly sets.
`partNames`	All	`Sequence[str]`	No	Current value	Switch the target to model parts.
`variables`	All	`Sequence[str] or None`	No	Current value	Replace the output-variable names; `None` clears them.
`location`	`PointProbe`	`Point3D`	No	Current value	Replace a point Probe's evaluation location.

For `LineProbe.setValues(*, name=..., instanceNames=..., setNames=..., partNames=..., variables=..., startLocation=..., stopLocation=..., numEvalPoints=...) -> None`:

Name	Applies to	Type	Required	Default	Description
`name`	All	`str or None`	No	Current value	May be omitted, empty, `None`, or unchanged; Probe renaming is not supported.
`instanceNames`	All	`Sequence[str]`	No	Current value	Switch the target to assembly instances.
`setNames`	All	`Sequence[str]`	No	Current value	Switch the target to assembly sets.
`partNames`	All	`Sequence[str]`	No	Current value	Switch the target to model parts.
`variables`	All	`Sequence[str] or None`	No	Current value	Replace the output-variable names; `None` clears them.
`startLocation`	`LineProbe`	`Point3D`	No	Current value	Replace a line Probe's first endpoint.
`stopLocation`	`LineProbe`	`Point3D`	No	Current value	Replace a line Probe's second endpoint.
`numEvalPoints`	`LineProbe`	`int`	No	Current value	Replace a line Probe's

Name	Applies to	Type	Required	Default	Description
					evaluation-point count.

For `MultiPointProbe.setValues(*, name=..., instanceNames=..., setNames=..., partNames=..., variables=..., locations=...) -> None`:

Name	Applies to	Type	Required	Default	Description
`name`	All	`str or None`	No	Current value	May be omitted, empty, `None`, or unchanged; Probe renaming is not supported.
`instanceNames`	All	`Sequence[str]`	No	Current value	Switch the target to assembly instances.
`setNames`	All	`Sequence[str]`	No	Current value	Switch the target to assembly sets.
`partNames`	All	`Sequence[str]`	No	Current value	Switch the target to model parts.
`variables`	All	`Sequence[str] or None`	No	Current value	Replace the output-variable names; `None` clears them.
`locations`	`MultiPointProbe`	`Sequence[Point3D]`	No	Current value	Replace a multi-point Probe's explicit locations.

Parameter rules

Rule	Requirement
Optional exclusive group	If updating this group, supply exactly one of `instanceNames`, `setNames`, or `partNames`.
Applicability	`location` applies only to `PointProbe`.
Applicability	`startLocation` applies only to `LineProbe`.
Applicability	`stopLocation` applies only to `LineProbe`.
Applicability	`numEvalPoints` applies only to `LineProbe`.
Applicability	`locations` applies only to `MultiPointProbe`.

Raises

Type	Condition
`ValidationError`	If an update is unsupported or produces an invalid Probe.

Example

```
>>> probe.setValues(setNames=("TIP",))
```