

Defines dispatching method

```
class TriangulateCell : public vtkm::worklet::WorkletMapPointToCell
{
public:
```

Defines how input arrays and structures are interpreted

```
    typedef void ControlSignature(CellSetIn topology,
                                ExecObject tables,
                                FieldOutCell<> connectivityOut);
```

```
    typedef void ExecutionSignature(CellShape, PointIndices, _2, _3, VisitIndex);
    using InputDomain = _1;
```

Specifies domain argument (optional)

```
    using ScatterType = vtkm::worklet::ScatterCounting;
    VTKM_CONT
```

```
    ScatterType GetScatter() const
    {
        return this->Scatter;
    }
```

Defines mapping from  
input domain to output  
domain (optional)

```
    template<typename CellShapeTag,
            typename ConnectivityInVec,
            typename ConnectivityOutVec>
```

```
    VTKM_EXEC
```

```
    void operator()(
        CellShapeTag shape,
        const ConnectivityInVec &connectivityIn,
        const internal::TriangulateTablesExecutionObject<DeviceAdapter> &tables,
        ConnectivityOutVec &connectivityOut,
        vtkm::IdComponent visitIndex) const
```

Defines how data are  
assigned to threads

```
{
```

Algorithms are just functions that  
run on a single instance of the input