Functions and subroutines defined in DefUtils.f90

```
FUNCTION GetSolver() RESULT( Solver )
   TYPE(Solver_t), POINTER :: Solver

FUNCTION GetMatrix( USolver ) RESULT( Matrix )
   TYPE(Matrix_t), POINTER :: Matrix
   TYPE(Solver_t), OPTIONAL, TARGET :: USolver

FUNCTION GetMesh( USolver ) RESULT( Mesh )
   TYPE(Mesh_t), POINTER :: Mesh
   TYPE(Solver_t), OPTIONAL, TARGET :: USolver

FUNCTION GetCurrentElement(Element) RESULT(Ret_Element)
  TYPE(Element_t), POINTER :: Ret_Element
  TYPE(Element_t), OPTIONAL, TARGET :: Element

FUNCTION GetElementIndex(Element) RESULT(Indx)
   TYPE(Element_t), OPTIONAL :: Element
   INTEGER :: Indx

FUNCTION GetNOFActive( USolver ) RESULT(n)
   INTEGER :: n
   TYPE(Solver_t), OPTIONAL, TARGET :: USolver

FUNCTION GetTime() RESULT(st)
   REAL(KIND=dp) :: st

FUNCTION GetTimeStep() RESULT(st)
```

```
   INTEGER :: st

FUNCTION GetTimeStepInterval() RESULT(st)
   INTEGER :: st

FUNCTION GetTimestepSize() RESULT(st)
   REAL(KIND=dp) :: st

FUNCTION GetCoupledIter() RESULT(st)
   INTEGER :: st

FUNCTION GetNonlinIter() RESULT(st)
   INTEGER :: st

FUNCTION GetNOFBoundaryElements( UMesh ) RESULT(n)
  INTEGER :: n
  TYPE(Mesh_t), OPTIONAL :: UMesh

SUBROUTINE GetScalarLocalSolution( x,name,UElement,USolver,tStep )
   REAL(KIND=dp) :: x(:)
   CHARACTER(LEN=*), OPTIONAL :: name
   TYPE(Solver_t)  , OPTIONAL, TARGET :: USolver
   TYPE(Element_t),  OPTIONAL, TARGET :: UElement
   INTEGER, OPTIONAL :: tStep

FUNCTION GetNofEigenModes( name,USolver) RESULT (NofEigenModes)
   CHARACTER(LEN=*), OPTIONAL :: name
   TYPE(Solver_t)  , OPTIONAL, TARGET :: USolver
   INTEGER :: NofEigenModes
```

```
FUNCTION GetString( List, Name, Found ) RESULT(str)
   TYPE(ValueList_t), POINTER :: List
   CHARACTER(LEN=*) :: Name
   LOGICAL, OPTIONAL :: Found
   CHARACTER(LEN=MAX_NAME_LEN) :: str

FUNCTION GetInteger( List, Name, Found ) RESULT(i)
   TYPE(ValueList_t), POINTER :: List
   CHARACTER(LEN=*) :: Name
   LOGICAL, OPTIONAL :: Found

FUNCTION GetLogical( List, Name, Found ) RESULT(l)
   TYPE(ValueList_t), POINTER :: List
   CHARACTER(LEN=*) :: Name
   LOGICAL, OPTIONAL :: Found

RECURSIVE FUNCTION GetConstReal( List, Name, Found,x,y,z ) RESULT(r)
   TYPE(ValueList_t), POINTER :: List
   CHARACTER(LEN=*) :: Name
   LOGICAL, OPTIONAL :: Found
   REAL(KIND=dp), OPTIONAL :: x,y,z
   REAL(KIND=dp) :: r

RECURSIVE FUNCTION GetCReal( List, Name, Found ) RESULT(s)
   TYPE(ValueList_t), POINTER :: List
   CHARACTER(LEN=*) :: Name
   LOGICAL, OPTIONAL :: Found
   REAL(KIND=dp) :: s
```

```
RECURSIVE FUNCTION GetReal( List, Name, Found, UElement ) RESULT(x)
     TYPE(ValueList_t), POINTER :: List
     CHARACTER(LEN=*) :: Name
     LOGICAL, OPTIONAL :: Found
     TYPE(Element_t), OPTIONAL, TARGET :: UElement
     REAL(KIND=dp), POINTER :: x(:)

  RECURSIVE FUNCTION GetParentMatProp( Name,UElement,Found,UParent ) &
                                       RESULT(x)
    CHARACTER(LEN=*) :: Name
    TYPE(Element_t), OPTIONAL, TARGET :: UElement
    LOGICAL, OPTIONAL :: Found
    TYPE(Element_t), OPTIONAL, POINTER :: UParent
    REAL(KIND=dp), POINTER :: x(:)

  FUNCTION GetElementProperty( Name, UElement ) RESULT(Values)
    CHARACTER(LEN=*) :: Name
    REAL(KIND=dp), POINTER :: Values(:)
    TYPE(Element_t), POINTER, OPTIONAL :: UElement

  FUNCTION GetActiveElement(t,USolver) RESULT(Element)
     INTEGER :: t
     TYPE(Element_t), POINTER :: Element
     TYPE( Solver_t ), OPTIONAL, TARGET :: USolver

  FUNCTION GetBoundaryElement(t,USolver) RESULT(Element)
     INTEGER :: t
     TYPE(Element_t), POINTER :: Element
```

```
   TYPE( Solver_t ), OPTIONAL, TARGET :: USolver

FUNCTION ActiveBoundaryElement(UElement,USolver) RESULT(l)
   TYPE(Element_t), OPTIONAL,  TARGET :: UElement
   TYPE(Solver_t),  OPTIONAL,  TARGET :: USolver
   LOGICAL :: l

FUNCTION GetElementCode( Element )  RESULT(etype)
   INTEGER :: etype
   TYPE(Element_t), OPTIONAL :: Element

FUNCTION GetElementFamily( Element )  RESULT(family)
   INTEGER :: family
   TYPE(Element_t), OPTIONAL :: Element
   TYPE(Element_t), POINTER :: CurrElement

FUNCTION GetElementNOFNodes( Element ) RESULT(n)
   INTEGER :: n
   TYPE(Element_t), OPTIONAL :: Element
   TYPE(Element_t), POINTER :: CurrElement

FUNCTION GetElementNOFDOFs( UElement,USolver ) RESULT(n)
   INTEGER :: n
   TYPE(Solver_t),  OPTIONAL, TARGET :: USolver
   TYPE(Element_t), OPTIONAL, TARGET :: UElement

FUNCTION GetElementDOFs( Indexes, UElement, USolver )  RESULT(NB)
   TYPE(Element_t), OPTIONAL, TARGET :: UElement
   TYPE(Solver_t),  OPTIONAL, TARGET :: USolver
   INTEGER :: Indexes(:)
   INTEGER :: NB

FUNCTION GetElementNOFBDOFs( Element, USolver ) RESULT(n)
  INTEGER :: n
  TYPE(Solver_t), OPTIONAL, POINTER :: USolver
  TYPE(Element_t), OPTIONAL :: Element

FUNCTION GetBodyForceId(  Element, Found ) RESULT(bf_id)
   LOGICAL, OPTIONAL :: Found
   TYPE(Element_t), OPTIONAL :: Element
   INTEGER :: bf_id

FUNCTION GetMaterialId( Element, Found ) RESULT(mat_id)
   LOGICAL, OPTIONAL :: Found
   TYPE(Element_t), OPTIONAL :: Element
   INTEGER :: mat_id

FUNCTION GetEquationId( Element, Found ) RESULT(eq_id)
   LOGICAL, OPTIONAL :: Found
   TYPE(Element_t), OPTIONAL :: Element
   INTEGER :: eq_id

FUNCTION GetSimulation() RESULT(Simulation)
   TYPE(ValueList_t), POINTER :: Simulation

FUNCTION GetConstants() RESULT(Constants)
   TYPE(ValueList_t), POINTER :: Constants
```

```
FUNCTION GetSolverParams(Solver) RESULT(SolverParam)
   TYPE(ValueList_t), POINTER :: SolverParam
   TYPE(Solver_t), OPTIONAL :: Solver

FUNCTION GetMaterial(  Element, Found ) RESULT(Material)
  TYPE(Element_t), OPTIONAL :: Element
  LOGICAL, OPTIONAL :: Found
  TYPE(ValueList_t), POINTER :: Material

FUNCTION GetBodyForce( Element, Found ) RESULT(BodyForce)
  TYPE(Element_t), OPTIONAL :: Element
  LOGICAL, OPTIONAL :: Found
  TYPE(ValueList_t), POINTER :: BodyForce

FUNCTION GetEquation( Element, Found ) RESULT(Equation)
  TYPE(Element_t), OPTIONAL :: Element
  LOGICAL, OPTIONAL :: Found
  TYPE(ValueList_t), POINTER :: Equation

FUNCTION GetBCId( UElement ) RESULT(bc_id)
   TYPE(Element_t), OPTIONAL, TARGET :: UElement
   INTEGER :: bc_id

FUNCTION GetBC( UElement ) RESULT(bc)
   TYPE(Element_t), OPTIONAL, TARGET :: UElement
   TYPE(ValueList_t), POINTER :: BC

FUNCTION GetICId( Element, Found ) RESULT(ic_id)
   LOGICAL, OPTIONAL :: Found
```

```
     TYPE(Element_t), OPTIONAL :: Element
     INTEGER :: ic_id,

  FUNCTION GetIC(  Element, Found ) RESULT(IC)
    TYPE(Element_t), OPTIONAL :: Element
    LOGICAL, OPTIONAL :: Found
    TYPE(ValueList_t), POINTER :: IC

  FUNCTION DefaultSolve( USolver ) RESULT(Norm)
    TYPE(Solver_t), OPTIONAL, TARGET :: USolver
    REAL(KIND=dp) :: Norm

   FUNCTION GaussPointsBoundary(Element, boundary, np) RESULT(gaussP)
     TYPE(Element_t) :: Element
     INTEGER, INTENT(IN) :: boundary, np
     TYPE( GaussIntegrationPoints_t ) :: gaussP

FUNCTION GetEdgeMap( ElementFamily ) RESULT(EdgeMap)
    INTEGER :: ElementFamily
    INTEGER, POINTER :: EdgeMap(:,:)

SUBROUTINE GetScalarLocalSolution( x,name,UElement,USolver,tStep )
     REAL(KIND=dp) :: x(:)
     CHARACTER(LEN=*), OPTIONAL :: name
     TYPE(Solver_t)  , OPTIONAL, TARGET :: USolver
     TYPE(Element_t),  OPTIONAL, TARGET :: UElement
     INTEGER, OPTIONAL :: tStep
  SUBROUTINE GetVectorLocalSolution( x,name,UElement,USolver,tStep )
     REAL(KIND=dp) :: x(:,:)
     CHARACTER(LEN=*), OPTIONAL :: name
     TYPE(Solver_t),  OPTIONAL, TARGET :: USolver
     TYPE(Element_t), OPTIONAL, TARGET :: UElement
     INTEGER, OPTIONAL :: tStep

SUBROUTINE GetScalarLocalEigenmode &
            ( x,name,UElement,USolver,NoEigen,ComplexPart )
     REAL(KIND=dp) :: x(:)
     CHARACTER(LEN=*), OPTIONAL :: name
     TYPE(Solver_t)  , OPTIONAL, TARGET :: USolver
     TYPE(Element_t),  OPTIONAL, TARGET :: UElement
     INTEGER, OPTIONAL :: NoEigen
     LOGICAL, OPTIONAL :: ComplexPart

  SUBROUTINE GetVectorLocalEigenmode &
        ( x,name,UElement,USolver,NoEigen,ComplexPart )
     REAL(KIND=dp) :: x(:,:)
     CHARACTER(LEN=*), OPTIONAL :: name
     TYPE(Solver_t),  OPTIONAL, TARGET :: USolver
     TYPE(Element_t), OPTIONAL, TARGET :: UElement
     INTEGER, OPTIONAL :: NoEigen
     LOGICAL, OPTIONAL :: ComplexPart

  RECURSIVE SUBROUTINE GetConstRealArray(List, x, Name, Found, UElement )
     TYPE(ValueList_t), POINTER :: List
     REAL(KIND=dp), POINTER :: x(:,:)
     CHARACTER(LEN=*) :: Name
```

```
   LOGICAL, OPTIONAL :: Found
   TYPE(Element_t), OPTIONAL, TARGET :: UElement

RECURSIVE SUBROUTINE GetRealArray( List, x, Name, Found, UElement )
   REAL(KIND=dp), POINTER :: x(:,:,:)
   TYPE(ValueList_t), POINTER :: List
   CHARACTER(LEN=*) :: Name
   LOGICAL, OPTIONAL :: Found
   TYPE(Element_t), OPTIONAL, TARGET :: UElement

SUBROUTINE SetElementProperty( Name, Values, UElement )
  CHARACTER(LEN=*) :: Name
  REAL(KIND=dp) :: Values(:)
  TYPE(Element_t), POINTER, OPTIONAL :: UElement

FUNCTION GetElementProperty( Name, UElement ) RESULT(Values)
  CHARACTER(LEN=*) :: Name
  REAL(KIND=dp), POINTER :: Values(:)
  TYPE(Element_t), POINTER, OPTIONAL :: UElement

SUBROUTINE GetElementNodes( ElementNodes, UElement, USolver )
   TYPE(Nodes_t) :: ElementNodes
   TYPE(Solver_t), OPTIONAL, TARGET :: USolver
   TYPE(Element_t), OPTIONAL, TARGET :: UElement

SUBROUTINE DefaultInitialize( Solver )
   TYPE(Solver_t), OPTIONAL :: Solver
```

```
SUBROUTINE DefaultDirichletBCs( USolver,Ux,UOffset )
   INTEGER, OPTIONAL :: UOffset
   TYPE(Variable_t), OPTIONAL, TARGET :: Ux
   TYPE(Solver_t), OPTIONAL, TARGET :: USolver

SUBROUTINE SolveLinSys( A, x, n )
   INTEGER :: n
   REAL(KIND=dp) :: A(n,n), x(n), b(n)

SUBROUTINE DefaultFinishAssembly( Solver )
  TYPE(Solver_t), OPTIONAL :: Solver

 FUNCTION GaussPointsBoundary(Element, boundary, np) RESULT(gaussP)
   TYPE(Element_t) :: Element
   INTEGER, INTENT(IN) :: boundary, np
   TYPE( GaussIntegrationPoints_t ) :: gaussP

 SUBROUTINE MapGaussPoints( Element, n, gaussP, Nodes )
   TYPE(Element_t) :: Element
   TYPE(GaussIntegrationPoints_t) :: gaussP
   TYPE(Nodes_t) :: Nodes
   INTEGER :: n

SUBROUTINE GetParentUVW( Element,n,Parent,np,U,V,W,Basis )
  TYPE(Element_t) :: Element, Parent
  INTEGER :: n, np
  REAL(KIND=dp) :: U,V,W,Basis(:)
```