**Tekla Open API – Создание основных объектов в модели.**

Давайте попробуем создать основные геометрические объекты (балка, колонна, фундамент и т.д.). Основное что нужно понять при создании элементов:

1. Если колонна отклоняется от вертикали на определенный угол она вставится как балка.
2. Железобетонная балка (колонна, стальная пластина и перекрытие, и т.д.) отличается от стальной балки (колонны, стальной пластины и перекрытия, и т.д.) тем, что у нее задан материал Бетон. Например B10: concreteBeam.Material.MaterialString = "B10"; По умолчанию элемент вставится как стальная балка.

/\*  
 \* Created by Vadim Semenov  
 \* vad.s.semenov@gmail.com  
 \* Date: 13.04.2021  
 \*   
 \*/  
**using** System;  
**using** Tekla.Structures.Model;  
**using** Tekla.Structures.Geometry3d;  
  
**namespace** CreateObjects  
{  
    class Program  
    {  
        **public** static void **Main**(string[] args)  
        {  
  
            **CreateSteelColumn**();  
            **CreateConcreteColumn**();  
            **CreateSteelBeam**();  
            **CreateConcreteBeam**();  
            **CreatePolyBeam**();  
            **CreateSpiralBeam**();  
            **CreateContourPlate**();  
            **CreatePadFooting**();  
            **CreateStripFooting**();  
            **CreateConcreteSlab**();  
            **CreatePanel**();  
                  
        }  
          
        //Стальная колонна  
        **private** static void **CreateSteelColumn**()  
        {  
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                      
                Beam steelColumn = **new** Beam();  
                steelColumn.Name = "Steel Column";  
                steelColumn.Profile.ProfileString = "I30K1\_20\_93";  
                steelColumn.Material.MaterialString = "C245";  
                steelColumn.StartPoint = **new** Point(0, 0, 0);  
                steelColumn.EndPoint.*Z* = 5000;  
                steelColumn.Position.Rotation = Position.**RotationEnum**.*FRONT*;  
                steelColumn.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                steelColumn.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                  
                steelColumn.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
        //Бетонная колонна  
        **private** static void **CreateConcreteColumn**()  
        {      
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                      
                Beam concreteColumn = **new** Beam();  
                concreteColumn.Name = "Concrete Column";  
                concreteColumn.Profile.ProfileString = "400\*400";  
                concreteColumn.Material.MaterialString = "B10";  
                concreteColumn.StartPoint = **new** Point(0, 6000, 0);  
                concreteColumn.EndPoint = **new** Point(0, 6000, 5000);  
                concreteColumn.Position.Rotation = Position.**RotationEnum**.*FRONT*;  
                concreteColumn.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                concreteColumn.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                  
                concreteColumn.**Insert**();  
      
                model.**CommitChanges**();  
            }  
                  
        }  
  
  
        //Стальная балка  
        **private** static void **CreateSteelBeam**()  
        {  
                          
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {          
                Beam steelBeam = **new** Beam();  
                steelBeam.Name = "Steel Beam";  
                steelBeam.Profile.ProfileString = "I30K1\_20\_93";  
                steelBeam.Material.MaterialString = "C245";  
                steelBeam.StartPoint = **new** Point(0, 0, 5000);  
                steelBeam.EndPoint = **new** Point(0, 6000, 5000);  
                steelBeam.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                steelBeam.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                steelBeam.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                  
                steelBeam.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
        //Бетонная балка  
        **private** static void **CreateConcreteBeam**()  
        {  
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                      
                Beam concreteBeam = **new** Beam();  
                concreteBeam.Name = "Concrete Beam";  
                concreteBeam.Profile.ProfileString = "400\*400";  
                concreteBeam.Material.MaterialString = "B10";  
                concreteBeam.StartPoint = **new** Point(0, 0, 5000);  
                concreteBeam.EndPoint = **new** Point(6000, 0, 5000);  
                concreteBeam.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                concreteBeam.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                concreteBeam.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                  
                concreteBeam.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
          
        //Составная балка  
        **private** static void **CreatePolyBeam**()  
        {  
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                  
                PolyBeam polyBeam = **new** PolyBeam();  
                polyBeam.Name = "PolyBeam";  
                polyBeam.Profile.ProfileString = "400\*400";  
                polyBeam.Material.MaterialString = "B15";  
                ContourPoint contourPoint1 = **new** ContourPoint();  
                ContourPoint contourPoint2 = **new** ContourPoint();  
                ContourPoint contourPoint3 = **new** ContourPoint();  
                  
                contourPoint1.**SetPoint**(**new** Point(0, 0, 0));  
                contourPoint2.**SetPoint**(**new** Point(6000, 6000, 0));  
                contourPoint3.**SetPoint**(**new** Point(12000, 6000, 0));  
                      
                polyBeam.**AddContourPoint**(contourPoint1);  
                polyBeam.**AddContourPoint**(contourPoint2);  
                polyBeam.**AddContourPoint**(contourPoint3);  
                  
                polyBeam.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                polyBeam.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                polyBeam.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                   
                polyBeam.**Insert**();  
                  
                model.**CommitChanges**();  
            }  
          
        }  
          
        //Спиральная балка  
        **private** static void **CreateSpiralBeam**()  
        {  
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                  
                SpiralBeam sb = **new** SpiralBeam(**new** Point(6000, 6000, 0), **new** Point(1000, 1000, 6000), 6000, 580);  
                  
                sb.Name = "PolyBeam";  
                sb.Profile.ProfileString = "400\*400";  
                sb.Material.MaterialString = "B15";  
                  
                sb.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                sb.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                sb.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                   
                sb.**Insert**();  
                  
                model.**CommitChanges**();  
            }  
          
        }  
          
          
        //Стальная пластина  
        **private** static void **CreateContourPlate**()  
        {  
                          
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {          
                  
                ContourPlate cp = **new** ContourPlate();  
                  
                cp.Name = "Contour Plate";  
                cp.Material.MaterialString = "C245";  
                cp.Profile.ProfileString = "PL10";  
                cp.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                cp.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                cp.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                
                ContourPoint contourPoint1 = **new** ContourPoint();  
                ContourPoint contourPoint2 = **new** ContourPoint();  
                ContourPoint contourPoint3 = **new** ContourPoint();  
                  
                contourPoint1.**SetPoint**(**new** Point(0, 0, 0));  
                contourPoint2.**SetPoint**(**new** Point(6000, 6000, 0));  
                contourPoint3.**SetPoint**(**new** Point(12000, 6000, 0));  
                      
                cp.**AddContourPoint**(contourPoint1);  
                cp.**AddContourPoint**(contourPoint2);  
                cp.**AddContourPoint**(contourPoint3);  
                  
                  
                cp.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
          
        //Перекрытие  
        **private** static void **CreateConcreteSlab**()  
        {  
                          
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {          
                  
                ContourPlate cp = **new** ContourPlate();  
                  
                cp.Name = "Concrete slab";  
                cp.Material.MaterialString = "B10";  
                cp.Profile.ProfileString = "160";  
                cp.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                cp.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                cp.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                
                ContourPoint contourPoint1 = **new** ContourPoint();  
                ContourPoint contourPoint2 = **new** ContourPoint();  
                ContourPoint contourPoint3 = **new** ContourPoint();  
                  
                contourPoint1.**SetPoint**(**new** Point(0, 0, 0));  
                contourPoint2.**SetPoint**(**new** Point(6000, 6000, 0));  
                contourPoint3.**SetPoint**(**new** Point(12000, 6000, 0));  
                      
                cp.**AddContourPoint**(contourPoint1);  
                cp.**AddContourPoint**(contourPoint2);  
                cp.**AddContourPoint**(contourPoint3);  
                  
                  
                cp.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
  
          
        //Блочный фундамент  
        **private** static void **CreatePadFooting**()  
        {  
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
          
                Beam pf = **new** Beam(Beam.**BeamTypeEnum**.*PAD\_FOOTING*);  
                pf.Name = "Pad footing";  
                pf.Profile.ProfileString = "1000\*1000";  
                pf.Material.MaterialString = "B10";  
                  
                pf.StartPoint.*X* = 20000;  
                pf.StartPoint.*Y* = 20000;  
                pf.EndPoint.*X* = 20000;  
                pf.EndPoint.*Y* = 20000;  
                pf.EndPoint.*Z* = -500;  
              
                pf.Position.Rotation = Position.**RotationEnum**.*FRONT*;  
                pf.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                pf.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                  
                pf.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
          
          
        //Ленточный фундамент  
        **private** static void **CreateStripFooting**()  
        {  
            Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                      
                Beam pf = **new** Beam(Beam.**BeamTypeEnum**.*STRIP\_FOOTING*);  
                pf.Name = "Pad footing";  
                pf.Profile.ProfileString = "5000\*1000";  
                pf.Material.MaterialString = "B10";  
                  
                pf.StartPoint.*X* = 20000;  
                pf.StartPoint.*Y* = 20000;  
                pf.EndPoint.*X* = 20000;  
                pf.EndPoint.*Y* = 20000;  
                pf.EndPoint.*Z* = -500;  
              
                pf.Position.Rotation = Position.**RotationEnum**.*FRONT*;  
                pf.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                pf.Position.Depth = Position.**DepthEnum**.*MIDDLE*;  
                  
                pf.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
          
          
        //Панель  
        **private** static void **CreatePanel**()  
        {  
                Model model = **new** Model();  
              
            **if** (model.**GetConnectionStatus**()) {  
                      
                Beam panel = **new** Beam(Beam.**BeamTypeEnum**.*PANEL*);  
                panel.Name = "Concrete Column";  
                panel.Profile.ProfileString = "2000\*200";  
                panel.Material.MaterialString = "B10";  
                panel.StartPoint = **new** Point(0, 6000, 0);  
                panel.EndPoint = **new** Point(6000,0 , 0);  
                panel.Position.Rotation = Position.**RotationEnum**.*TOP*;  
                panel.Position.Plane = Position.**PlaneEnum**.*MIDDLE*;  
                panel.Position.Depth = Position.**DepthEnum**.*FRONT*;  
                  
                panel.**Insert**();  
      
                model.**CommitChanges**();  
            }  
        }  
          
          
          
    }  
}