ENTERPRISE ARCHITECT

Série de Guides d'Utilisateur

Graphiques dynamiques

Author: Sparx Systems
Date: 23/11/2023
Version: 16.1

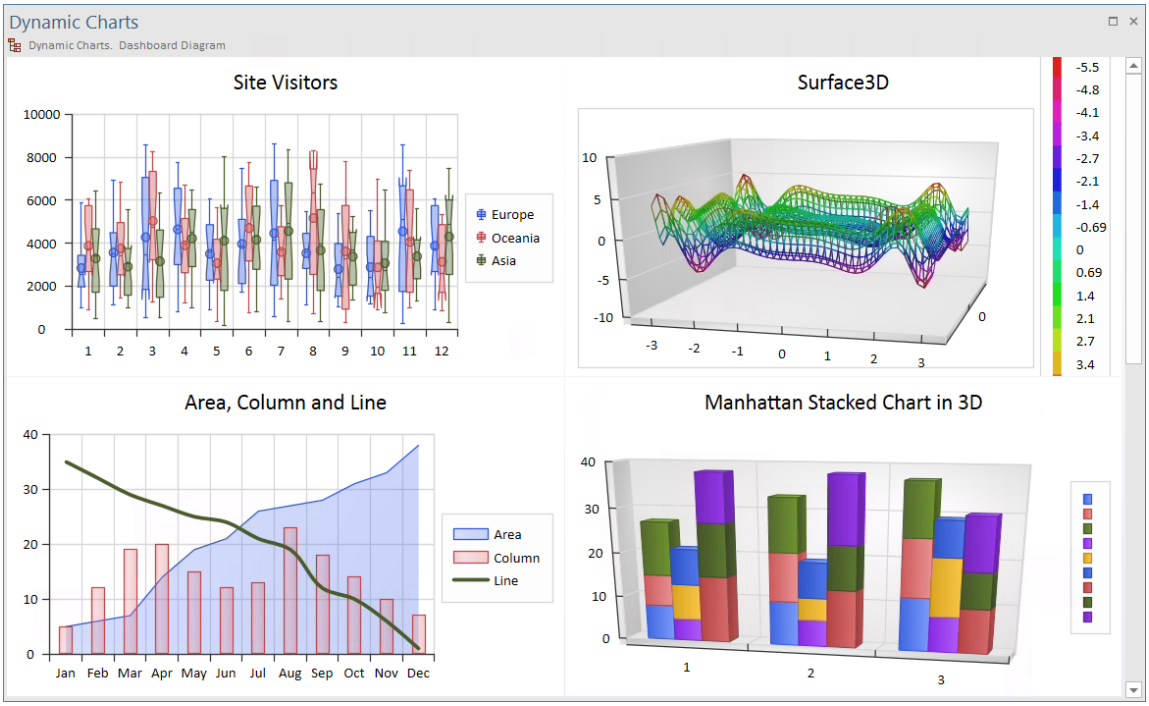
CRÉÉ AVEC ENTERPRISE ARCHITECT

Table des Matières

Graphiques dynamiques 3
- Définition de graphique à l'aide de JSON 4
- Définition de graphique à l'aide de Simulation 7
- Définition de graphique à l'aide de JavaScript 10
- Ressources de graphiques dynamiques 15
- L'API Graphique 16
- Classe de graphique 17
- Énumérations de graphique 20
  - ChartAxisCrossType 21
  - ChartAxisIndex 22
  - ChartAxisLabelType 23
  - ChartAxisTickMarkTypeChartAxisTickMarkType 24
  - Type d'axe de graphique 25
  - FormeBarreGraphique 26
  - CatégorieGraphique 27
  - ChartColorMode 29
  - ChartCurveType 30
  - Style de tiret de graphique 31
  - Style de cadre de graphique 32
  - ChartGradientTypeChartGradientType 33
  - GraphiqueMarkerShape 34
  - GraphiqueStockSeriesType 35
  - Type de graphique 36
  - ChartWallOptions 37
- Classe ChartAxisIndexChartAxisIndex Class 38
- Classe ChartDataValueChartDataValue Class 40
- Classe ChartDiagram3DChartDiagram3D Class 41
- Classe ChartFormatSeriesChartFormatSeries Class 42
- Classe ChartSeries 43

Graphiques dynamiques

fonctionnalités Enterprise Architect Artefacts DynamicChart, qui sont des éléments de graphique qui peuvent être stylisés et rendus dynamiquement à l'ouverture du diagramme . Les graphiques dynamiques s'appuient entièrement sur du code pour définir leurs séries, leur style et leur contenu et sont entièrement gérés via l'interface d'automatisation par les clients - généralement des plug-ins et des scripts. Les interfaces Chart fournissent aux clients les moyens de décrire et de remplir dynamiquement un graphique lorsque le diagramme est affiché. L'API est vaste et flexible, vous permettant d'illustrer graphiquement de nombreux scénarios différents au moment de l'exécution. La fonctionnalité vous permet de définir tout type de graphique dont vous avez besoin en utilisant JavaScript sous forme de code ou de JSON.



Une bonne référence pour les interfaces Chart et leur utilisation est le Paquetage ' Rapportage > Charts > Dynamic Charts' dans l' Enterprise Architect Exemple Modèle . Ce Paquetage contient des exemples de nombreux graphiques, chacun décrit dynamiquement par les clients JavaScript . Chaque exemple de graphique présente les deux méthodes de rendu : une méthode de codage JavaScript et une méthode de source de données JSON.

Les graphiques dynamiques sont particulièrement utiles pour représenter les résultats des simulations, vous permettant de :

- Enregistrez les résultats de votre Simulation sous forme d'éléments visuels du graphique
- Incluez facilement des graphiques remplis par les résultats Simulation dans vos rapports
- Partagez des résultats Simulation conviviaux avec les parties prenantes sans avoir besoin d'outils Simulation supplémentaires

Les graphiques dynamiques sont disponibles dans les éditions Corporate , Unified et Ultimate d' Enterprise Architect .

Définition de graphique à l'aide de JSON

Plutôt que de coder vous-même un graphique dynamique, vous souhaiterez peut-être fournir une simple description du graphique. Les graphiques dynamiques peuvent être conçus et entièrement définis par une seule source de données ; JSON est actuellement le format de source de données préféré, mais XML et d'autres seront disponibles à l'avenir.

Vous définissez le graphique en fournissant une structure de données JSON simple qui adhère au schéma DynamicChart. Le schéma est disponible dans le Compositeur de Schéma . Il est également facilement visible dans le Paquetage de graphiques dynamiques du Modèle d'exemple Enterprise Architect .

(Pour afficher le schéma DynamicChart, sélectionnez Développer > Modélisation de schéma > Compositeur de Schéma > Ouvrir Compositeur de Schéma , cliquez sur le bouton ... dans le champ « Profil » et sélectionnez DynamicChartSchema.)

Sources de données - JSON

Pour afficher un graphique à l'aide d'une structure de données JSON, sélectionnez d'abord l'élément DynamicChart Artifact, puis ouvrez l'éditeur « code interne ». Pour une sélection dans le Navigateur , cliquez-droit et choisissez ' Fonctionnalités > Editer le code interne' ou, pour une sélection sur un diagramme , cliquez-droit et choisissez 'Editer le script du graphique'. Ceux-ci ouvriront l'éditeur pour que vous puissiez modifier le script Chart. Créez une variable JSON qui définit le graphique à restituer, puis composez votre fonction ConstructChart.

La fonction ConstructChart prend comme argument unique l'identité (une string GUID) de l'élément Chart affiché sur le diagramme d'ouverture. Vous appelez ensuite la fonction intégrée ConstructChartFromJSON, en passant le GUID comme premier paramètre et la structure JSON comme deuxième argument, comme illustré dans cet exemple :

Example Datasource in JSON

```
var barChart2DJSON =
{
 "Category" : "BarSmart",
 "Type" : "Simple",
 "Title" : "Vehicle Expenses",
 "Series" :
 [
   {
                "Label" : "Fuel",
                "Data":
                {
                        "Type" : "Column",
                        "Points" :
                        [
                                { "Category": "Jan", "Y": 1.0 },
                                { "Category": "Feb", "Y": 3.0 },
                                { "Category": "Mar", "Y": 7.0 },
                                { "Category": "Apr", "Y": 8.0 },
                                { "Category": "May", "Y": 10.0 },
```

```
                { "Category": "Jun", "Y": 15.0 }
            ]
        }
    },
    {
        "Label" : "Taxes",
        "Data":
        {
            "Type" : "Normal",
            "Points" :
            [
                { "Y":10.0 },
                { "Y":12.0 },
                { "Y":16.0 },
                { "Y":17.0 },
                { "Y":10.0 },
                { "Y":12.0 }
            ]
        }
    },
    {
        "Label" : "Maintenance",
        "Data":
        {
            "Type" : "Normal",
            "Points" :
            [
                { "Y":5.0 },
                { "Y":2.0 },
                { "Y":6.0 },
                { "Y":7.0 },
                { "Y":1.0 },
                { "Y":2.0 }
            ]
        }
    },
    {
        "Label" : "Other",
        "Data":
        {
            "Type" : "Normal",
            "Points" :
```

```
                    [
                        { "Y":2.5 },
                        { "Y":2.5 },
                        { "Y":2.5 },
                        { "Y":2.5 },
                        { "Y":2.5 },
                        { "Y":2.5 }
                    ]
                }
            }
    ]
};

function ConstructChart(chartGuid)
{
        ConstructChartFromJSON(chartGuid, barChart2DJSON);
}
```

Autres exemples

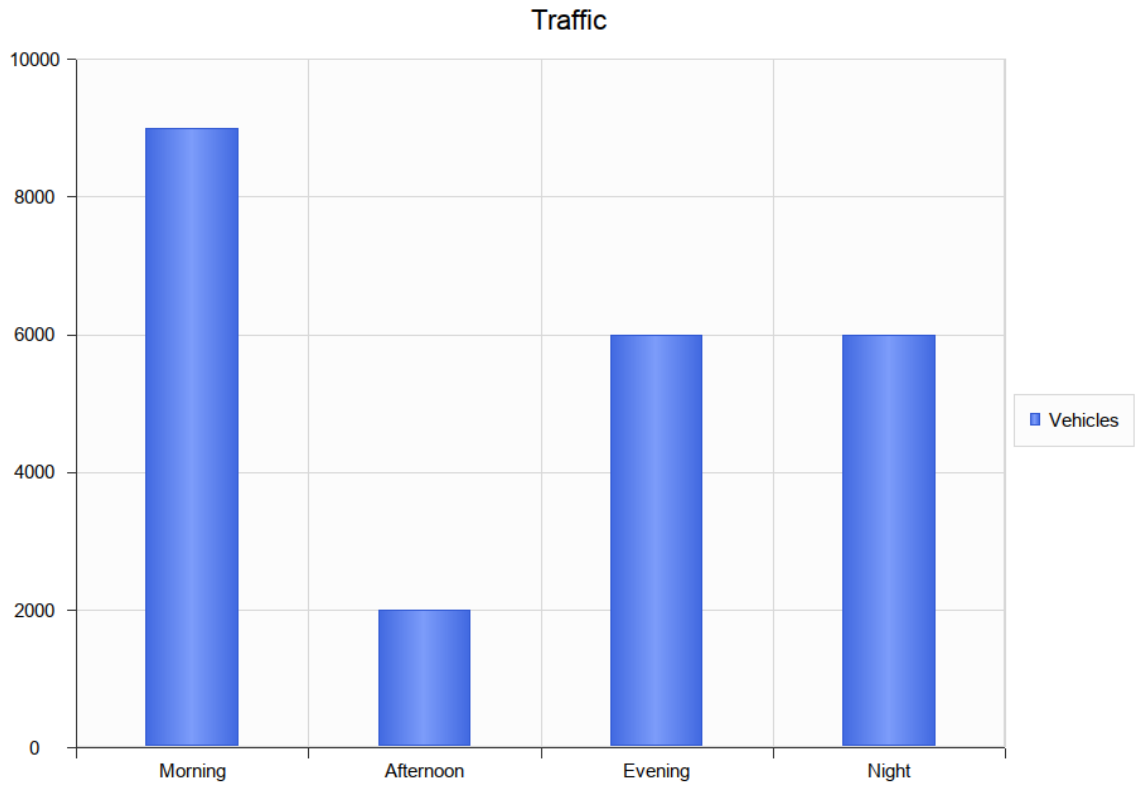
D'autres exemples JSON sont fournis dans l'Exemple Modèle (voir le Paquetage ' Rapportage > Charts > Dynamic Charts').

Chaque exemple de graphique fournit un diagramme de tableau de bord et un élément DynamicChart. Sélectionnez un élément parmi l'un de ces exemples et appuyez sur « Alt+7 » pour afficher le comportement derrière le graphique. Examiner la variété d'exemples de graphiques est le meilleur moyen de savoir comment utiliser JSON pour produire les types de graphiques qui pourraient vous intéresser.

Définition de graphique à l'aide de Simulation

Les simulations sont un outil fantastique pour observer les comportements. À tout moment de la Simulation il est facile de savoir où nous nous trouvons et dans quel état nous nous trouvons. Au fur et à mesure que nous progressons dans une Simulation , ces informations sont généralement ignorées. Dans une Simulation qui nous montre, par exemple, les chiffres du trafic sur une période de 24 heures, nous pourrions facilement observer le nombre de véhicules circulant dans un tunnel à différentes heures du matin et du soir. Il peut être utile de conserver ces informations une fois la Simulation terminée et de les utiliser pour fournir quelque chose de significatif. La fonctionnalité DynamicChart dans Simulation nous permet de faire exactement cela. En prenant l'exemple ci-dessus, nous pourrions enregistrer la quantité de trafic à chaque étape de la Simulation et l'utiliser pour produire un graphique qui montrerait clairement la quantité de trafic qui a traversé le tunnel pendant la période de 24 heures de la Simulation . Les graphiques peuvent en effet nous afficher soit la chronologie d'une Simulation , soit l'effet total d'une simulation.

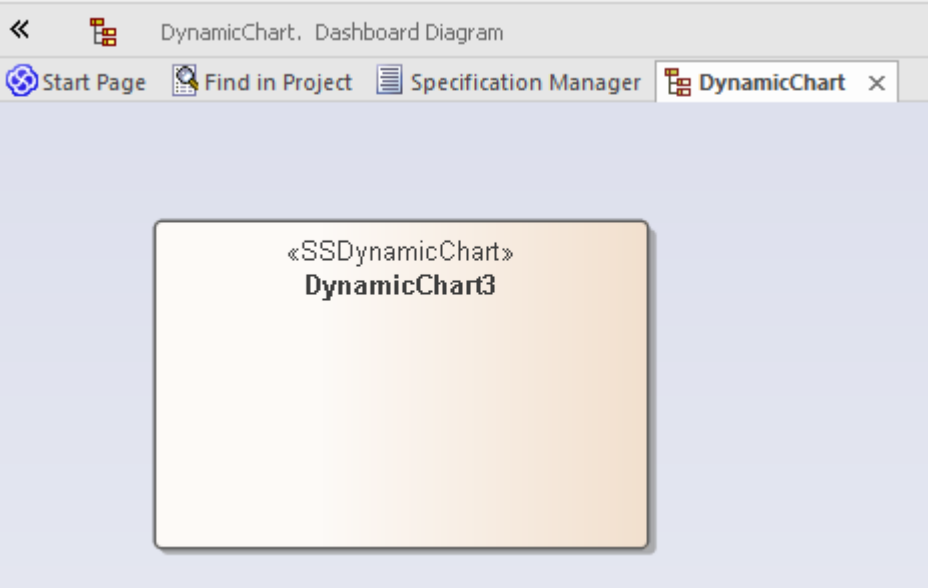
Producing Custom Charts in Simulation



You can fashion all sorts of Charts from any Simulation. Each time a Simulation is run, any DynamicChart elements referenced (by name) by the Simulation are updated. The Simulation will search for any named Chart in the same Package as the model.

Follow this simple process:

1. Create a DynamicChart element for the Simulation.



2. In the initial step of the Simulation, use JavaScript to define a variable to hold the vehicle numbers.

```
//
//  the traffic variable will hold the traffic numbers as Simulation proceeds and is initially zero
//  each element of the array represents a period of the day, morning, afternoon, evening and night.
//
var traffic = [0,0,0,0];
```

3. Next write the JavaScript that describes, in JSON format, the Chart to produce.

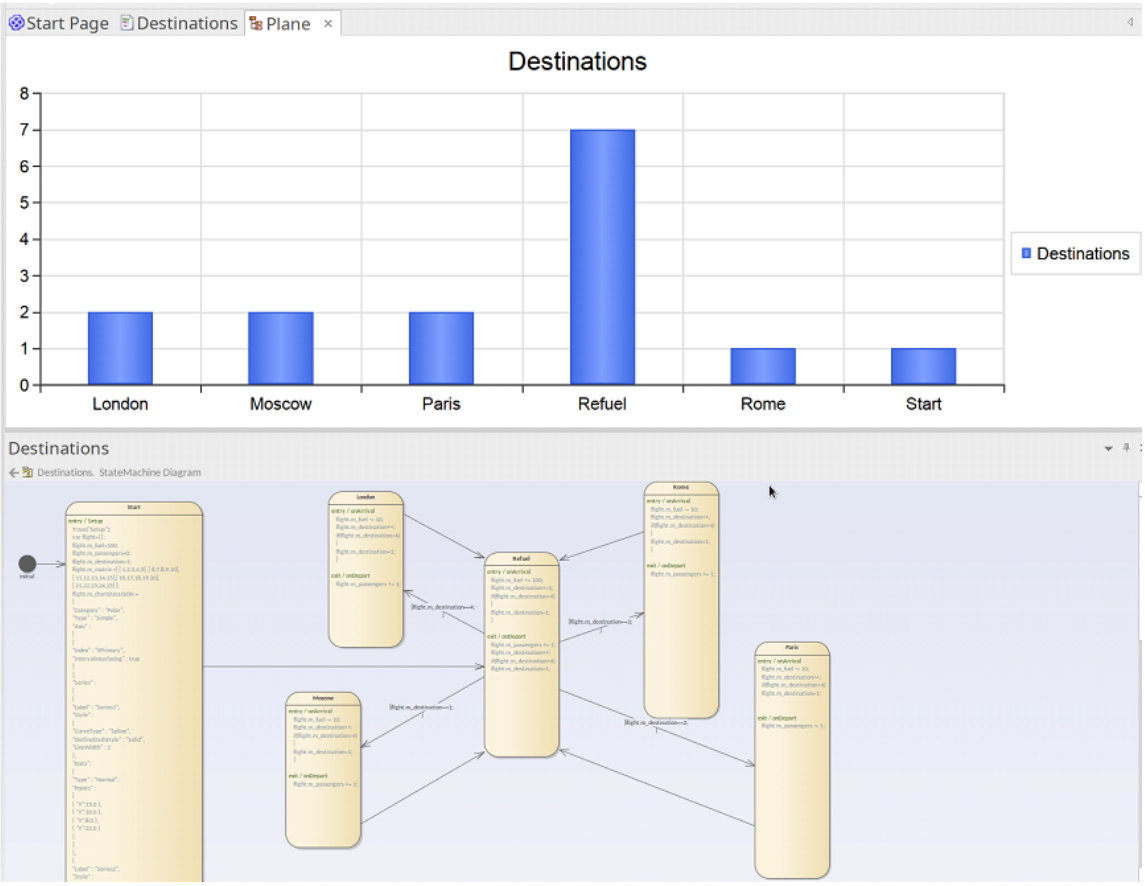
```
//
// The JSON instance describing the chart to produce. (complies with the EA DynamicChart Schema)
//
    var chartData =
    {
     "Category" : "Column",
      "Type" : "Simple",
      "Title" : "Traffic",
      "Series" :
      [
            {       "Label" : "Vehicles",
                    "Data" :
                    {
                            "Type" : "Column",
                            "Points" :
                            [
                                    { "Category": "Morning", "Y" : 0 },//  The Y values of the axis are initially
zero
                                    { "Category": "Afternoon", "Y" : 0 },        //  they will be filled in at end
of Simulation
                                    { "Category": "Evening", "Y" : 0 },
                                    { "Category": "Night", "Y" : 0 }
                            ]
                    }
            }
      ]
    };
```

4. At various transitions in the Simulation update the traffic numbers.

```
//
// 2000 vehicles went through the tunnel in the afternoon (element 1)
//
traffic[1] += 2000;
```

5. At the end of the Simulation, use the data captured during the run to fill the series.
```
// fill points in series with the number of vehicles for each part of the day
var dataPoints = chartData.Series[0].Data.Points;
for(var dp = 0; dp < traffic.length; dp++)
{
    dataPoints[dp].Y = traffic[dp];
}
```
6. Update the model.
```
// Call the EA function to populate the DynamicChart element named 'Vehicles' with this data.
sim.GenerateChart( "Vehicles", JSON.stringify(chartData));
```

Graphique par défaut produit par Simulation



En plus des graphiques que vous produisez spécifiquement, un graphique récapitulatif peut être produit automatiquement par une simulation. Il suffit d'ajouter un DynamicChart Artifact au Paquetage et de lui donner le même nom que le Statemachine . Le graphique par défaut résume les transitions State effectuées lors de l'exécution d'une simulation. Si un graphique par défaut est trouvé une fois la simulation terminée, les données de ce graphique sont mises à jour et affichées automatiquement.

Pour ajouter un graphique par défaut à votre simulation Statemachine , procédez comme suit :

1. Localisez le Paquetage contenant la Statemachine sur laquelle effectuer la simulation.
2. Créez un diagramme de tableau de bord en tant qu'enfant de ce Paquetage .
3. Ajoutez un artefact DynamicChart au diagramme du tableau de bord et donnez-lui le même nom que Statemachine .

Une fois la simulation terminée, ouvrez simplement le diagramme tableau de bord pour révéler le résumé.

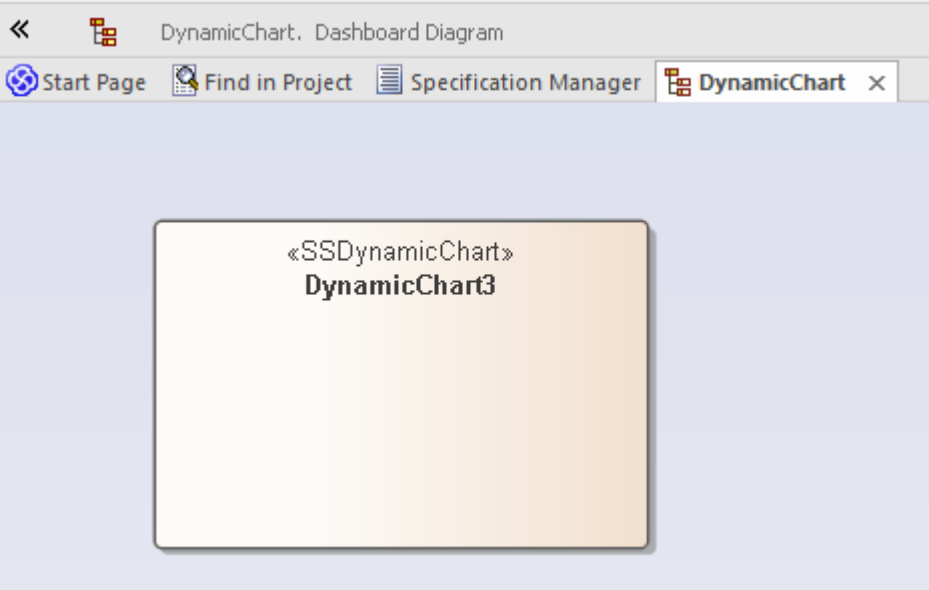
Définition de graphique à l'aide de JavaScript

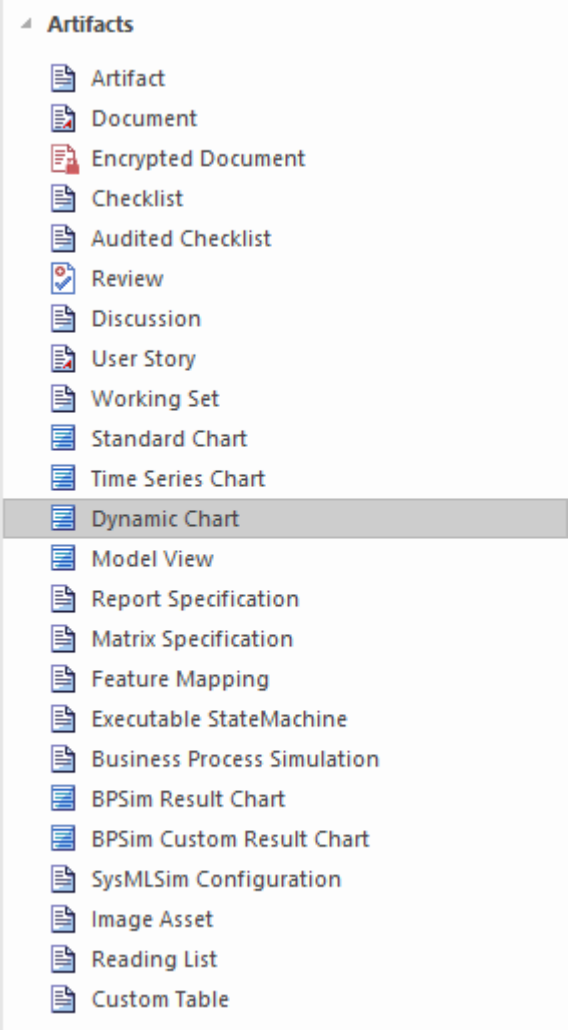
Dans cette rubrique, nous discutons du codage d'un artefact DynamicChart en utilisant uniquement JavaScript et l'interface d'automatisation Chart.

Définir un graphique via JavaScript

La première chose à faire est de créer un diagramme de tableau de bord dans le Paquetage approprié. Cliquez-droit sur le Paquetage et sélectionnez l'option 'Ajouter Diagramme '.

Dans la dialogue ' Nouveau Diagramme ', sélectionnez le type 'Construction > Diagrammes et graphiques' et, lorsque le diagramme vide s'affiche, faites glisser l'icône 'Dynamic Chart' dessus depuis la page 'Graphiques' de la Boîte à outils.





Vous écrivez maintenant le JavaScript pour styliser et restituer le graphique, en commençant par la fonction ConstructChart qui sera invoquée automatiquement chaque fois que le diagramme contenant l'artefact DynamicChart est ouvert pour visualisation. Le GUID de l'élément est transmis à ConstructChart en tant que paramètre. Dans cette fonction, c'est à vous de décider quel type de graphique afficher, le style du graphique, le nombre de séries qu'il contient et les points de données qui composent la série. En utilisant le Paquetage de graphiques de l'interface d'automatisation, il est possible d'afficher presque tous les graphiques dont vous avez besoin.

Dans cet exemple, vous allez créer un graphique à colonnes groupées qui montre les dépenses liées aux véhicules sur quelques mois. Chaque groupe représentera un mois et sera décomposé selon les différentes dépenses engagées au cours de ce mois.

Pour commencer, cliquez sur l'artefact et appuyez sur Alt+7, ou cliquez sur l'option de menu contextuel « Modifier le script de graphique » ; chaque méthode affiche la fenêtre Éditeur de Code . Le code à utiliser est fourni ici, suivi du graphique qu'il produira à l'ouverture du diagramme .

note surtout :

- Le **!INC Locale Scripts .ChartAutomation** déclaration; tous les scripts Chart doivent inclure cette instruction
- La fonction ConstructChart (7ème ligne)

Code

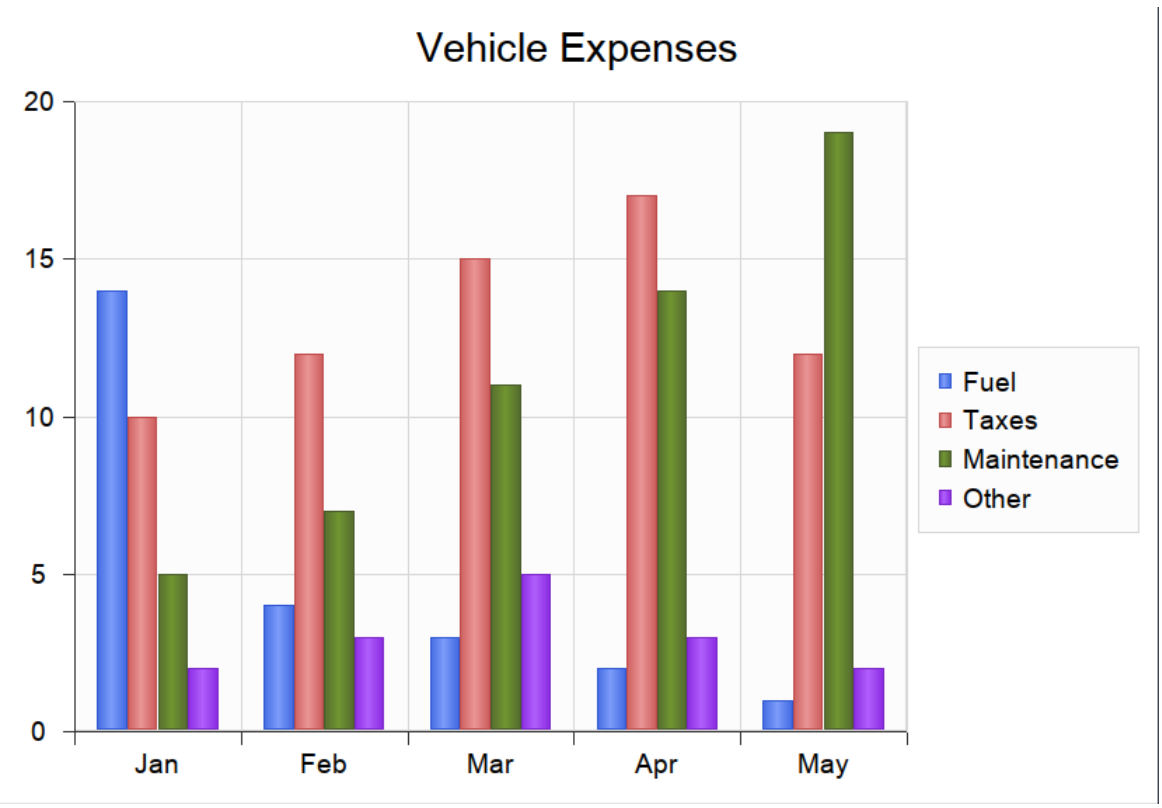
!INC Local Scripts.ChartAutomation

```
var monthNames = [ "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec" ];

function Rand(min, max) {
  min = cephes.ceil(min);
  max = cephes.floor(max);
  return cephes.floor(cephes.drand() * (max - min)) + min; }

function ConstructChart( guid )
{
        var chart as EA.Chart;                          // The script first of all
        var element = GetElementByGuid(guid);                // declares the automation
        var series1 as EA.ChartSeries;                   // objects it will use
        var series2 as EA.ChartSeries;
        var series3 as EA.ChartSeries;
        var series4 as EA.ChartSeries;

        chart = element.GetChart();

        var chartCategory = ChartCategory.Column();
        var chartType = ChartType.SIMPLE();
        chart.SetChartType( chartCategory, chartType, false, true);

        chart.Title = "Vehicle Expenses";
        series1 = chart.CreateSeries("Fuel");          //  The script then obtains the Chart object and creates the
        series2 = chart.CreateSeries("Taxes");          //  series. A chart is composed of a number of series, and
        series3 = chart.CreateSeries("Maintenance");    //  in this example each series will represent a type of expense.
        series4 = chart.CreateSeries("Other");

        series1.AddDataPoint3( monthNames[0], 14);   //  A series is composed of a number of datapoints and, here, the

        series1.AddDataPoint3( monthNames[1], 4);    //  script adds the values for each of the points to each series.
        series1.AddDataPoint3( monthNames[2], 3);
        series1.AddDataPoint3( monthNames[3], 2);
        series1.AddDataPoint3( monthNames[4], 1);

        series2.AddDataPoint(10);
        series2.AddDataPoint(12);
        series2.AddDataPoint(15);
        series2.AddDataPoint(17);
        series2.AddDataPoint(12);
```

```
        series3.AddDataPoint(5);
        series3.AddDataPoint(7);
        series3.AddDataPoint(11);
        series3.AddDataPoint(14);
        series3.AddDataPoint(19);

        series4.AddDataPoint(2);
        series4.AddDataPoint(3);
        series4.AddDataPoint(5);
        series4.AddDataPoint(3);
        series4.AddDataPoint(2);

        series1.SetGroupID(0);
        series1.SetGroupID(0);
        series3.SetGroupID(1);
        series4.SetGroupID(1);

        chart.Redraw();
}
```

Sortir

Il s'agit du graphique produit par le code.



Débogage d'un graphique dynamique

Après avoir créé Dynamic Chart JavaScript , vous pouvez le déboguer comme pour n'importe quel autre code. Cliquez-droit sur le Graphique Dynamique dans un diagramme et sélectionnez l'option ' Déboguer Chart Script'. Le script s'affiche dans le Déboguer Vue .

Autres exemples

D'autres exemples de codage sont fournis dans l'Exemple Modèle (voir le Paquetage ' Rapportage > Charts > Dynamic Charts'). Chaque exemple de graphique fournit un diagramme de tableau de bord et un élément DynamicChart. Sélectionnez l'un de ces éléments et appuyez sur Alt+7 pour afficher le code de comportement derrière le graphique. Se référer à ces exemples est le meilleur moyen de comprendre comment coder chaque type de graphique.

JavaScript est le langage principal pour coder des graphiques dynamiques à l'aide de l'automatisation. Cependant, il est certainement possible qu'un client d'automatisation tiers soit impliqué dans ce processus, l'hôte JavaScript déléguant des tâches aux clients d'automatisation dans des langages tels que C# et C++, qui pourraient être en mesure d'obtenir des données en dehors du modèle de différentes manières. non disponible pour les scripts.

Ressources de graphiques dynamiques

Un certain nombre de ressources sont à votre disposition pour vous aider à créer des graphiques de manière dynamique dans Enterprise Architect , comme décrit dans le tableau *Ressources* .

Ressources

Ressource	Description
Bibliothèque JavaScript	Une bibliothèque de fonctions JavaScript , avec des exemples de codage de graphiques utilisant l'interface d'automatisation ou utilisant JSON comme source de données pour les graphiques. Cette bibliothèque se trouve dans le groupe 'Local Scripts ' du contrôle Scriptant dans Enterprise Architect .
Exemple Modèle	Le Modèle d'exemple Enterprise Architect contient un certain nombre d'exemples de graphiques, illustrant les différents types de graphiques qui peuvent être créés et stylisés de manière dynamique lorsqu'ils sont affichés. Ceux-ci se trouvent dans le Paquetage « Rapportage > Graphiques > Graphiques dynamiques ». Les exemples incluent des graphiques de surface et filaires 3D, des graphiques boursiers avec des séries montrant les fluctuations de prix et de volume, des graphiques en boîte à moustaches montrant le nombre de visiteurs, des graphiques Manhattan Stack et bien d'autres. Chaque exemple est fourni en deux versions. La première version utilise du code JavaScript pour styliser et remplir le graphique. La seconde utilise une source de données JSON pour décrire et remplir le graphique.
Schémas de graphiques	L'exemple Modèle contient un modèle de référence pour les graphiques dynamiques, comprenant des diagrammes de classes, un profil Compositeur de Schéma et les schémas JSON et XML générés à l'aide du profil, sous forme d'artefacts que vous pouvez visualiser rapidement en double-cliquant dessus.

L'API Graphique

L'interface Chart est l' object API qui fournit des méthodes pour créer dynamiquement des graphiques. Il peut être utilisé pour construire n'importe lequel des types de graphiques pris en charge.

Une interface Chart est obtenue à l'aide de la méthode GetChart sur un élément DynamicChart. Un élément DynamicChart peut être créé à partir de la page « Graphiques » de la boîte à outils Diagramme et est généralement utilisé sur un diagramme tableau de bord.

Classe de graphique

La classe Chart est l'interface principale pour les éléments Chart ; il est utilisé pour créer une série, ajouter des points de données à une série et configurer l'apparence du graphique.

Attributes du graphique

Attribute	Description
Titre	String Notes : lecture/écriture Le titre du graphique.
Catégorie	CatégorieGraphique Notes : Lecture seule La catégorie graphique ; fourni dans la méthode SetChartType.
Type	Type de graphique Notes : lecture seule Le type de graphique ; fourni dans la méthode SetChartType.

Graphique Méthodes

Method	Description
AddChartDataYXZ(double Y, double X, double Z, longue sérieIndex)	long Ajoute un point de données à une série existante. Paramètres: • Y : double, la valeur de l'axe Y primaire • X : double, la valeur de l'axe X primaire • Z : double, la valeur de l'axe Z primaire • seriesIndex : long, l'index de la série (retourné par les méthodes CreateSeries)
AddChartDataYY1 (catégorie string , double Y, double Y1, longue sérieIndex)	long Ajoute un point de données à une série existante. Paramètres: • catégorie : string - le groupe, la colonne ou l'étiquette de l'axe x • Y : double, la valeur de l'axe Y primaire • Y1 : double, la valeur de l'axe Y secondaire • seriesIndex : long, l'index de la série (retourné par les méthodes CreateSeries et CreateSeriesEx)
CreateSeries(nom string)	LDISPATCH

	Ajoute une nouvelle série au graphique. Renvoie une interface qui peut être utilisée pour ajouter des données à la série et configurer son apparence. Paramètres: • name : string , le nom affiché de la série
CreateSeriesEx (nom string , couleur longue, type ChartType, catégorie ChartCategory)	LDISPATCH Crée une série d'une catégorie et d'un type de graphique particulier et renvoie une interface IChartSeries. Cela permet aux graphiques de former plusieurs séries de différentes manières. Les CombinedCharts du Modèle EAExample en sont un exemple, affichant respectivement trois séries pour les catégories Area, Column et Line. Paramètres: • name: string , le nom de la série • couleur : long, valeur de couleur RVB ,-1 par défaut • type : ChartType, l'une des énumérations ChartType • catégorie : ChartCategory, l'une des énumérations ChartCategory
EnableResizeAxes (booléen bEnable)	annuler Accorde ou refuse la possibilité de redimensionner les axes. Paramètres: • bEnable : booléen
GetChartAxis (type ChartAxisType)	LDISPATCH Retourne une interface IChartAxis pour l'axe spécifié. Paramètres: • type : ChartAxisType, l'une des énumérations ChartAxisType
ObtenirDiagramme3D()	LDISPATCH Retourne une interface IChartDiagram qui peut être utilisée pour spécifier le moteur de rendu ; Logiciel ou OpenGL.
GetSeries (index long)	LDISPATCH Retourne une interface IChartSeries pour l'index donné. Les indices des séries commencent à zéro. Paramètres: • indice : long
GetSeriesCount()	long Renvoie le nombre de séries représentées par le graphique.
Redessiner()	annuler Redessine le graphique.
SetChartType (type ChartType, catégorie ChartCategory, booléen bRedraw, booléen bResizeAxis)	annuler Il s'agit généralement du premier appel effectué sur l'interface graphique. Il définit le style et l'apparence du graphique lorsqu'il est rendu. Paramètres: • type : type de graphique • catégorie : Catégorie de graphique

	• bRedessiner : booléen • bResizeAxis : booléen
SetCurveType (type ChartCurveType)	annuler Ensembles la méthode d'interpolation pour tracer la courbe. Paramètres: • type : ChartCurveType, l'une des énumérations ChartCurveType, telles que Line, Spline ou SplineHermite.
SetSeriesShadow(boolean bShow)	annuler Affiche ou masque les ombres sur les séries. Paramètres: • bShow : booléen
SetThemeOpacity (pourcentage long)	annuler Ensembles l'opacité du graphique. Paramètres: • pourcentage : long, transparence du graphique en pourcentage
ShowAxis (index long)	annuler Affiche l'axe pour l'index donné. Paramètres: • index : long, une des énumérations ChartAxisType
ShowDataLabels(boolean show, boolean border, boolean dropLineTomarker)	annuler Affiche ou masque les étiquettes de données sur le graphique. Paramètres: • show : booléen, afficher ou masquer les étiquettes • bordure : booléen, affiche ou masque la bordure sur les étiquettes • dropLineTomarker : booléen, modifie la position de l'étiquette par rapport à la ligne
ShowDataMarkers (spectacle booléen, taille longue, forme ChartmarkerShape)	annuler Affiche ou masque les marqueurs de données sur le graphique. Permet également de définir l'apparence des marqueurs. Paramètres: • show : booléen, afficher ou masquer les marqueurs • taille : long, taille des marqueurs en pixels • forme : ChartmarkerShape, l'une des énumérations ChartmarkerShape

Énumérations de graphique

Ces énumérations, utilisées spécifiquement par les méthodes de l'interface Chart, sont décrites dans les rubriques de cette section. Cliquez sur le nom de l'énumération dans la liste à gauche de ce texte.

ChartAxisCrossType

Valeurs d'énumération

Enum	Value
Auto	valeur : 0
MaximumAxisValue	valeur : 1
MinimumAxisValue	valeur : 2
ValeurAxe	valeur : 3
Ignorer	valeur : 4
FixedDefaultPos	valeur : 5

ChartAxisIndex

Valeurs d'énumération

Enum	Value
Inconnu	valeur : -1
X	valeur : 0
Oui	valeur : 1
Z	valeur : 2

ChartAxisLabelType

Valeurs d'énumération

Enum	Value
Aucun libellé	valeur : 0
NextToAxis	valeur : 1
Haut	valeur : 2
Faible	valeur : 3

ChartAxisTickMarkTypeChartAxisTickMarkType

Valeurs d'énumération

Enum	Value
Pas de tiques	valeur : 0
À l'intérieur	valeur : 1
Dehors	valeur : 2
Croix	valeur : 3

Type d'axe de graphique

Ensemble de constantes faisant référence aux différents axes utilisés dans les graphiques.

Valeurs d'énumération

Enum	Value
CHART_Y_PRIMARY_AXIS	valeur : 0
CHART_Y_SECONDARY_AXIS	valeur : 1
CHART_X_PRIMARY_AXIS	valeur : 2
CHART_X_SECONDARY_AXIS	valeur : 3
CHART_Z_PRIMARY_AXIS	valeur : 4
CHART_Z_SECONDARY_AXIS	valeur : 5
CHART_Y_POLAR_AXIS	valeur : 6
CHART_X_POLAR_AXIS	valeur : 7
CHART_A_TERNARY_AXIS	valeur : 8
CHART_B_TERNARY_AXIS	valeur : 9
CHART_C_TERNARY_AXIS	valeur : 10

FormeBarreGraphique

Valeurs d'énumération

Enum	Value
Boîte	valeur : 0
Pyramide	valeur : 1
PyramidePartielle	valeur : 2

CatégorieGraphique

Valeurs d'énumération

Enum	Value
graphiqueDéfaut	valeur : 0
graphiqueLigne	valeur : 1
graphiqueCartet	valeur : 2
graphiquePie3D	valeur : 3
graphiquePyramide	valeur : 4
graphiquePyramide3D	valeur : 5
graphiqueEntonnoir	valeur : 6
graphiqueEntonnoir3D	valeur : 7
graphiqueColonne	valeur : 8
graphiqueBar	valeur : 9
graphiqueHistogramme	valeur : 10
zone de graphique	valeur : 11
graphiqueStock	valeur : 12
graphiquebulle	valeur : 13
graphiqueLongData	valeur : 14
graphiqueLigneHistorique	valeur : 15
graphiquePolaire	valeur : 16
graphiqueDonut	valeur : 17
graphiqueDonut3D	valeur : 18
graphiqueTorus3D	valeur : 19
graphiqueTernaire	valeur : 20
graphiqueColonne3D	valeur : 21

graphiqueBarre3D	valeur : 22
graphiqueLigne3D	valeur : 23
chartArea3D	valeur : 24
graphiqueSurface3D	valeur : 25
graphiqueDonutImbriqué	valeur : 26
graphiqueBoxPlot	valeur : 27
graphiqueBarSmart	valeur : 28
graphiqueBar3DSmart	valeur : 29

ChartColorMode

Valeurs d'énumération

Enum	Values
Seul	valeur : 0
Plusieurs	valeur : 1
Palette	valeur : 2
Coutume	valeur : 3
Série	valeur : 4

ChartCurveType

Valeurs d'énumération

Enum	Value
Pas de ligne	valeur : 0
Doubler	valeur : 1
Spline	valeur : 2
SplineHermite	valeur : 3
Marcher	valeur : 4
Étape inversée	valeur : 5

Style de tiret de graphique

Valeurs d'énumération

Enum	Value
Solide	valeur : 0
Se précipiter	valeur : 1
Point	valeur : 2
TiretPoint	valeur : 3
TiretPointPoint	valeur : 4
Coutume	valeur : 5

Style de cadre de graphique

Valeurs d'énumération

Enum	Value
Aucun	valeur : 0
Engrener	valeur : 1
Contour	valeur : 2
ContourMesh	valeur : 3

ChartGradientTypeChartGradientType

Valeurs d'énumération

Enum	Value
Aucun	valeur : 0
Horizontal	valeur : 1
Vertical	valeur : 2
DiagonaleGauche	valeur : 3
Diagonale Droite	valeur : 4
CenterHorizontal	valeur : 5
CentreVertical	valeur : 6
RadialTop	valeur : 7
RadialCentre	valeur : 8
Fond radial	valeur : 9
RadialGauche	valeur : 10
RadialDroite	valeur : 11
RadialHautGauche	valeur : 12
RadialHautDroite	valeur : 13
RadialBasGauche	valeur : 14
RadialBasDroite	valeur : 15
Biseau	valeur : 16
TuyauVertical	valeur : 17
TuyauHorizontal	valeur : 18

GraphiqueMarkerShape

Valeurs d'énumération

Enum	Value
Cercle	valeur : 0
Triangle	valeur : 1
Rectangle	valeur : 2
Rhombe	valeur : 3

GraphiqueStockSeriesType

Valeurs d'énumération

Enum	Value
Bar	valeur : 0
Bougie	valeur : 1
LigneOuvrir	valeur : 2
LigneHaut	valeur : 3
LigneBas	valeur : 4
LigneFermer	valeur : 5
LignePersonnalisée	valeur : 6

Type de graphique

Valeurs d'énumération

Enum	Value
type de graphiqueDEFAULT	valeur : 0
type de graphique SIMPLE	valeur : 1
type de graphiqueSTACKED	valeur : 2
chartType100STACKED	valeur : 3
type de graphique PLAGE	valeur : 4

ChartWallOptions

Valeurs d'énumération

Enum	Value
Aucun	valeur : 0, 0x0000
FillLeftWall	valeur : 1, 0x0001
ContourGaucheMur	valeur : 2, 0x0002
FillRightWall	valeur : 4, 0x0004
ContourRightWall	valeur : 8, 0x0008
RemplirPlancher	valeur : 16, 0x0010
OutlineFloor	valeur : 32, 0x0020
Dessinertout	valeur : 65535, 0xFFFF
DessinerMurGauche	FillLeftWall | ContourGaucheMur
DrawRightWall	FillRightWall | ContourRightWall
DessinerPlancher	RemplirPlancher | OutlineFloor
DessinerTousLesMurs	DrawLeftWall | DrawRightWall
OutlineAllWalls	ContourGaucheMur | ContourRightWall
OutlineAll	OutlineAllWalls | OutlineFloor
RemplirTousLes Murs	FillLeftWall | FillRightWall
Remplir tout	RemplirTousLesMurs | RemplirPlancher
Défaut	OutlineAll

Classe ChartAxisIndexChartAxisIndex Class

Attributes ChartAxisIndex

Attribute	Description
Visible	booléen Affiche ou masque l'axe.

Méthodes ChartAxisIndex

Method	Description
EnableMajorUnitIntervalInterlacing (entrelacement booléen)	annuler Active ou désactive l'entrelacement.
GetGuid()	string Renvoie le guide de l'axe. Identifie un axe de manière unique.
GetLabel()	string Renvoie la valeur du label de l'axe.
SetAxisName (étiquette string , booléen showonaxis)	annuler Ensembles le libellé de l'axe et indique s'il doit être affiché sur le graphique. Paramètres: • label : string , le texte de l'étiquette • showonaxis : booléen, une valeur vraie indique que l'étiquette est affichée
SetCrossType (type long)	annuler Fournit une directive ou un conseil à utiliser lors du calcul de la position des étiquettes sur un axe. Paramètres: • type : long, une des énumérations ChartAxisCrossType
SetDataFormat(format string , format booléenAsDate)	annuler Ensembles de la string de format pour la conversion des valeurs en chaînes (par exemple "%.4f"). Si les points de données représentent des valeurs datetime, l'argument formatAsDate doit être vrai et la string de format définie de manière appropriée (par exemple "%H:%M") Paramètres: • format : string , le format à utiliser lors de la conversion des valeurs de point de données en string • formatAsDate : booléen, une valeur vraie indique que le point de données représente une date/heure

SetDisplayUnits (unités doubles)	annuler Ensembles les unités d'affichage sur l'axe. Fondamentalement, les valeurs des points de données sont divisées par ce chiffre pour donner une valeur d'unité majeure. Par exemple, si le point de données contient des valeurs en mètres, une valeur de 1000 entraînerait l'utilisation des kilomètres comme unité principale sur l'axe. Paramètres: • unités : double, la valeur d'une seule unité sur l'axe
SetFixedDisplayRange(double fmin, double fmax)	annuler Ensembles une plage fixe pour l'axe. Paramètres: • fmin : double, la valeur minimale • fmax : double, la valeur maximale
SetLabelType(long labelpos)	annuler Ensembles la position des étiquettes sur l'axe. Paramètres: • labelpos : long, l'une des énumérations ChartAxisLabelType
SetTickMark(long tickmarkpos)	annuler Ensembles la position des graduations sur l'axe. Paramètres: • tickmarkpos : long, l'une des énumérations ChartAxisTickMarkType
ShowMajorGridLines (boolean show)	annuler Affiche ou masque les lignes de la grille.

Classe ChartDataValueChartDataValue Class

La classe ChartDataValue fournit une interface qui permet d'obtenir des valeurs à partir de points d'une série.

Méthodes ChartDataValue

Method	Description
ObtenirValeur()	double Renvoie la valeur associée au point de données.
IsEmpty()	booléen Vrai si aucune valeur n'existe pour le point de données.
SetEmpty(booléen vide)	annuler Ensembles un point de données sur une série pour qu'il soit vide. Paramètres: • vide : booléen, vrai si le point de données doit être considéré comme vide, n'ayant pas valeur
SetValue(double valeur)	annuler Ensembles la valeur d'un point de données. Paramètres: • valeur : double, la valeur du point de données ; définir une valeur rend un point de données non vide

Classe ChartDiagram3DChartDiagram3D Class

Méthodes ChartDiagram3D

Method	Description
SetDrawWallOptions (options longues, rafraîchissement booléen)	annuler Ensembles l'option pour la façon dont les murs et les sols - le cas échéant - sont affichés sur le graphique 3D. Le paramètre options est un masque binaire d'une ou plusieurs valeurs de l'énumération ChartWallOptions. Paramètres: • options : Long, bitmask des options d'affichage au mur et floor • redraw : booléen, redessine le graphique une fois la fonction terminée
SetRenderingType (moteur long)	annuler Paramètres: • moteur : long, 0 pour logiciel, 1 pour openGL

Classe ChartFormatSeriesChartFormatSeries Class

Une classe d'assistance pour la classe ChartSeries qui permet de définir des options d'apparence.

Méthodes ChartFormatSeries

Method	Description
SetCurveType (type ChartCurveType)	annuler Ensembles l'option graphique pour le rendu des lignes. Paramètres: • type : long, une des énumérations ChartCurveType
SetSeriesLineWidth (largeur longue)	annuler Ensembles la largeur de ligne en pixels. Paramètres: • largeur : long, une valeur de pixel
SetSeriesOutlineDashStyle (ChartDashStyle dashstyle)	annuler Ensembles le style de tiret de la ligne sur le graphique/graphique. Paramètres: • dashstyle : ChartDashStyle, l'une des énumérations ChartDashStyle

Classe ChartSeries

ChartSeries Méthodes

Method	Description
AddBoxPlotData(double ave, double min, double q1, double q2, double q3, double max, double encoche)	long Pour un graphique ayant la catégorie BoxPlot, ajoute un seul point de données à la série. Paramètres: • ave : double, la valeur moyenne à ce point • min : double, la valeur minimale à ce stade • q1 : double, la valeur du premier quartile • q2 : double, la valeur du deuxième quartile • q3 : double, la valeur du troisième quartile • max : double de la valeur maximale à ce stade • notched : double, pour une série de style notched, la valeur notched à exprimer à cet endroit
Ajouter un point de données (double Y)	long Ajoute un point de données à la série. Renvoie l'index du point, qui est le nombre de points -1. Paramètres: • Y : double, la valeur de l'axe Y
AddDataPoint2(double Y, double X)	long Ajoute un point de données à la série. Renvoie l'index du point, qui est le nombre de points -1. Paramètres: • Y : double, la valeur de l'axe Y • X : double, la valeur de l'axe X
AddDataPoint3(catégorie string , double Y)	long Ajoute une valeur d'axe Y pour une catégorie donnée sur l'axe X. Paramètres: • category : string , le nom de la catégorie ou de la colonne • Y : double, la valeur
AddStockData (double ouverture, double haut, double bas, double fermeture, horodatage VARIANT)	annuler Ajoute des données à une série pour un graphique de la catégorie Stock. Paramètres: • ouvert : double, valeur d'ouverture • haut : double, haute valeur • bas : double, faible valeur • fermeture : double, valeur de fermeture • timestamp : {datetime, double utcsecs} soit VARIANT date valeur soit double,

	auquel cas la valeur est interprétée comme le nombre de secondes depuis minuit le 1er janvier 1970, heure UTC
AddSurfaceColors (VARIANTES de couleurs)	annuler Ajoute une ou plusieurs couleurs à la série. Paramètres: • couleurs : long, ou long[], une seule couleur RVB ou un tableau de valeurs de couleurs RVB
CloseShape (fermeture booléenne, remplissage booléen)	annuler Relie les premier et dernier points de données et remplit la forme si 'fill' est vrai. Paramètres • close : booléen, si vrai ferme la série • fill : booléen, remplit la forme
GetDataPointCount()	long Renvoie le nombre de points de données dans la série.
GetDataPointValue (index long)	LDISPATCH Renvoie une interface ChartDataValue pour le point de données avec l'index donné. Paramètres: • index : long, l'index du point de données (généralement renvoyé par les fonctions AddDataPoint ; une valeur comprise entre 0 et n-1, où n est le nombre de points renvoyés par la fonction *GetDataPointCount*)
GetSeriesFormat()	LDISPATCH Renvoie une interface ChartFormatSeries qui permet de modifier l'apparence du graphique.
SetBarShape (forme de barre longue)	annuler Ensembles la forme pour les graphiques à barres, 0 pour Box, 1 pour Pyramid, 2 pour PyramidPartial. Paramètres: • barshape : ChartBarShape, l'une des énumérations ChartBarShape
SetColorMapCount (nombre long)	annuler Ensembles le nombre de couleurs utilisées lors du rendu de la série. Les valeurs typiques sont 4, 8, 16 et 32 Paramètres: • count : long, le nombre de couleurs à utiliser
SetColorMode (mode ChartColorMode)	annuler Pour les graphiques 3D, définit la méthode d'interpolation pour le remplissage des formes. Simple, par exemple, entraînerait le remplissage de object 3D en variant légèrement la couleur. Le niveau de variation dépendra du nombre de couleurs utilisées par le graphique (voir *SetColorMapCount*). Paramètres: • mode : ChartColorMode

SetDrawFlat(plat booléen)	annuler Dessine la forme aplatie lorsqu'elle est définie sur true. Paramètres: • plat : booléen, dessin à plat
SetFrameColor (couleur longue)	annuler Ensembles la couleur du cadre pour les objets 3D. Paramètres: • color : long, la valeur de couleur RVB pour colorer le cadre
SetFrameStyle (style ChartFrameStyle)	annuler Ensembles le style de cadre pour le graphique - aucun, maillage, contour ou les deux. Paramètres: • style : ChartFrameStyle, l'une des énumérations ChartFrameStyle
SetGradientType (type long)	annuler Ensembles le type de dégradé à utiliser. Paramètres: • type : long, une des énumérations ChartGradientType
SetGroupID (identifiant long)	annuler Regroupe les séries sur un graphique empilé ayant le même identifiant. Doit être un nombre non négatif. Paramètres: • id : long, un nombre non négatif utilisé pour regrouper les séries sur un graphique
SetLevelRangeMode (mode long)	annuler Ensembles le mode pour les gammes en série. • 0 - Minimum et maximum pour la série • 1 - Minimum et maximum pour l'axe Y • 2 - Personnalisé Paramètres: • mode : long, 0 ou 1 pris en charge
SetRelatedAxis(axe string , index long)	annuler Ensembles l'axe associé pour une série. L'axe associé est créé à l'aide de la fonction Split de l'interface ChartAxis. L'axe est d'abord créé à l'aide de Split, puis une nouvelle série est créée, et cette fonction l'appelle sur l'un de ses axes. L'axe est spécifié par le paramètre index ; la valeur est l'une des énumérations ChartAxisIndex (0 pour X, 1 pour Y ou 2 pour Z) Paramètres: • axis : string , le guid de l'axe renvoyé par un appel de méthode ChartAxis.Split ; renvoyé par la méthode ChartAxis.GetGuid • index : long, une des énumérations ChartAxisIndex
SetStockSeriesType(TypeChartStockSeriesType)	annuler

	Pour les graphiques boursiers, définit le graphique utilisé pour afficher la série. Paramètres: • type : ChartStockSeriesType, l'une des énumérations ChartStockSeriesType
SetWireFrame (booléen câblé)	annuler Ensembles ou désactive l'option filaire. Lorsqu'il est défini sur true, le graphique n'est plus rendu sous la forme d'un object solide, mais sous la forme d'un cadre composé de fils. Paramètres: • wired : booléen, s'affiche sous la forme d'un object filaire si vrai