

## Drilling down into data

One common use of charts is to allow the user to drill down into the data. This usually occurs when the user performs some sort of event on the chart such as clicking on a wedge in a `PieChart` control or clicking on a column in a `ColumnChart` control. Clicking on a data item reveals a new chart that describes the make-up of that data item.

For example, you might have a `ColumnChart` control that shows the month-by-month production of widgets. To initially populate this chart, you might make a database call (through a service or some other adapter). If the user then clicks on the January column, the application could display the number of widgets of each color that were produced that month. To get the individual month's widget data, you typically make another database call and pass a parameter to the listening service that describes the specific data you want. You can then use the resulting data provider to render the new view.

The most common method of providing drill-down functionality is to make calls that are external to the application to get the drill-down data. You typically do this by using the chart's `itemClick` event listener, which gives you access to the `HitData` object. The `HitData` object lets you examine what data was underneath the mouse when the event was triggered. This capability lets you perform actions on specific chart data. For more information, see “Using the `HitData` object” on page 1745.

You can also use a simple `Event` object to get a reference to the series that was clicked. The following example shows the net worth of a fictional person. When you click on a column in the initial view, the example drills down into a second view that shows the change in value of a particular asset class over time.

The example uses the Event object to get a reference to the clicked ColumnSeries. It then drills down into the single ColumnSeries by replacing the chart's series Array with the single ColumnSeries in the chart. When you click a column again, the chart returns to its original configuration with all ColumnSeries.

```
<?xml version="1.0"?>
<!-- charts/SimpleDrillDown.mxml -->
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" height="100%"
width="100%" creationComplete="initApp()">
  <mx:Script><![CDATA[
    import mx.collections.ArrayCollection;

    [Bindable]
    public var dpac:ArrayCollection = new ArrayCollection ([
      { date:"01/01/2006", cash:50000,
        stocks:198192, retirement:130101,
        home:750000, other:19148 },
      { date:"02/01/2006", cash:50000,
        stocks:210309, retirement:143707,
        home:760000, other:19493 },
      { date:"03/01/2006", cash:50000,
        stocks:238992, retirement:169529,
        home:770000, other:19933 },
      { date:"04/01/2006", cash:50000,
        stocks:292269, retirement:242596,
        home:770000, other:21445 }]);

    public var initSeriesArray:Array = new Array();
    public var level:Number = 1;
    public var newSeries:Array;

    private function initApp():void {
      // Get initial series Array -- to be reloaded when it returns
      // from a drill down.
      initSeriesArray = chart.series;
    }

    private function zoomIntoSeries(e:Event):void {
      newSeries = new Array();
      if (level == 1) {
        newSeries.push(e.currentTarget);
        level = 2;
      } else {
        newSeries = initSeriesArray;
        pl.title = "Net Worth";
        level = 1;
      }
      chart.series = newSeries;
    }
  ]]></mx:Script>
</mx:Application>
```

```

]]></mx:Script>

<mx:Panel id="p1" title="Net Worth">
  <mx:ColumnChart id="chart"
    dataProvider="{dpac}"
    type="stacked"
    showDataTips="true"
  >
    <mx:series>
      <mx:ColumnSeries id="s1"
        displayName="Cash"
        yField="cash"
        xField="date"
        click="zoomIntoSeries(event)"
      />
      <mx:ColumnSeries id="s2"
        displayName="Stocks"
        yField="stocks"
        xField="date"
        click="zoomIntoSeries(event)"
      />
      <mx:ColumnSeries id="s3"
        displayName="Retirement"
        yField="retirement"
        xField="date"
        click="zoomIntoSeries(event)"
      />
      <mx:ColumnSeries id="s4"
        displayName="Home"
        yField="home"
        xField="date"
        click="zoomIntoSeries(event)"
      />
      <mx:ColumnSeries id="s5"
        displayName="Other"
        yField="other"
        xField="date"
        click="zoomIntoSeries(event)"
      />
    </mx:series>
    <mx:horizontalAxis >
      <mx:DateTimeAxis title="Date" dataUnits="months"/>
    </mx:horizontalAxis>
  </mx:ColumnChart>
  <mx:Legend dataProvider="{chart}"/>
</mx:Panel>
</mx:Application>

```

Another approach to drilling down into chart data is to use unused data within the existing data provider. You can do this by changing the properties of the series and axes when the chart is clicked.

The following example is similar to the previous example in that it drills down into the assets of a fictional person's net worth. In this case, though, it shows the value of the asset classes for the clicked-on month in the drill-down view rather than the change over time of a particular asset class.

This example uses the `HitData` object's `item` property to access the values of the current data provider. By building an `Array` of objects with the newly-discovered data, the chart is able to drill down into the series without making calls to any external services.

Drilling down into data is an ideal time to use effects such as `SeriesSlide`. This example also defines `seriesIn` and `seriesOut` effects to slide the columns in and out when the drilling down (and the return from drilling down) occurs.

```
<?xml version="1.0"?>
<!-- charts/DrillDownWithEffects.mxml -->
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" height="100%"
width="100%">
    <mx:Script><![CDATA[
        import mx.collections.ArrayCollection;
        import mx.charts.HitData;
        import mx.charts.events.ChartItemEvent;

        [Bindable]
        public var dpac:ArrayCollection = new ArrayCollection ([
            { date:"01/01/2006", assets:1607441, cash:520000, stocks:98192,
              retirement:130101, home:850000, other:9148 },
            { date:"02/01/2006", assets:1610509, cash:520000, stocks:97309,
              retirement:133707, home:850000, other:9493 },
            { date:"03/01/2006", assets:1617454, cash:520000, stocks:97992,
              retirement:139529, home:850000, other:9933 },
            { date:"04/01/2006", assets:1615310, cash:520000, stocks:92269,
              retirement:142596, home:850000, other:10445 },
            { date:"05/01/2006", assets:1600304, cash:520000, stocks:80754,
              retirement:139029, home:850000, other:10521 },
            { date:"06/01/2006", assets:1600416, cash:520000, stocks:80667,
              retirement:139024, home:850000, other:10725 },
            { date:"07/01/2006", assets:1599340, cash:520000, stocks:78913,
              retirement:139265, home:850000, other:11162 },
            { date:"08/01/2006", assets:1608965, cash:520000, stocks:84754,
              retirement:142618, home:850000, other:11593 },
            { date:"09/01/2006", assets:1622719, cash:520000, stocks:91078,
              retirement:149257, home:850000, other:12384 },
            { date:"10/01/2006", assets:1629806, cash:520000, stocks:86452,
              retirement:160310, home:850000, other:13044 },
            { date:"11/01/2006", assets:1642285, cash:520000, stocks:92172,
              retirement:166357, home:850000, other:13756 },
            { date:"12/01/2006", assets:1651009, cash:520000, stocks:95095,
              retirement:171557, home:850000, other:14357 }]);

        [Bindable]
        public var drillDownDataSet:ArrayCollection;

        [Bindable]
        public var dp:ArrayCollection = dpac;

        private function zoomIntoSeries(e:ChartItemEvent):void {
            if (dp==dpac) {
                drillDownDataSet = new ArrayCollection(genData(e));
                cs1.displayName = "Assets";
            }
        }
    ]]></mx:Script>
</mx:Application>
```

```

        cs1.yField = "amount";
        cs1.xField = "type";

        ca1.categoryField = "type";

        pl.title = "Asset breakdown for " + e.hitData.item.date;
        dp = drillDownDataSet;

        // Remove the Legend. It is not needed in this view because
        // each asset class is its own column in the drill down.
        pl.removeChild(myLegend);
    } else {
        cs1.displayName = "All Assets";
        cs1.yField = "assets";
        cs1.xField = "date";

        ca1.categoryField = "date";

        pl.title = "Net Worth";
        dp = dpac;

        // Add the Legend back to the Panel.
        pl.addChild(myLegend);
    }
}

private function genData(e:ChartItemEvent):Array {
    var result:Array = [];

    trace("Cash: " + e.hitData.item.cash);
    trace("Stocks: " + e.hitData.item.stocks);
    trace("Retirement: " + e.hitData.item.retirement);
    trace("Home: " + e.hitData.item.home);
    trace("Other: " + e.hitData.item.other);

    var o1:Object = {
        type:"cash",
        amount:e.hitData.item.cash
    };
    var o2:Object = {
        type:"stocks",
        amount:e.hitData.item.stocks
    };
    var o3:Object = {
        type:"retirement",
        amount:e.hitData.item.retirement
    };
    var o4:Object = {
        type:"home",
        amount:e.hitData.item.home
    };
}

```

```

    };
    var o5:Object = {
        type:"other",
        amount:e.hitData.item.other
    };
    trace(o1.type + ":" + o1.amount);

    result.push(o1);
    result.push(o2);
    result.push(o3);
    result.push(o4);
    result.push(o5);
    return result;
}
]]</mx:Script>

<mx:SeriesSlide id="slideIn" duration="1000" direction="down"/>
<mx:SeriesSlide id="slideOut" duration="1000" direction="up"/>

<mx:Panel id="p1" title="Net Worth">
    <mx:ColumnChart id="chart"
        showDataTips="true"
        itemClick="zoomIntoSeries(event)"
        dataProvider="{dp}"
    >
        <mx:series>
            <mx:ColumnSeries id="cs1"
                displayName="All Assets"
                yField="assets"
                xField="date"
                hideDataEffect="slideOut"
                showDataEffect="slideIn"/>
        </mx:series>

        <mx:horizontalAxis>
            <mx:CategoryAxis id="ca1" categoryField="date"/>
        </mx:horizontalAxis>
    </mx:ColumnChart>

    <mx:Legend id="myLegend" dataProvider="{chart}"/>
</mx:Panel>
</mx:Application>

```

