What's New in Maple 2016

Maple

DataSeries and DataFrame

Maple 2016 adds two new data containers: DataSeries and DataFrame. These labeled tabular data structures are ideal for storage of many different kinds of data:

- Tabular data with heterogeneous columns data types
- Ordered or unordered data, including time series or sequential data
- Any kind of statistical or observational data; labels are not essential for the data frame

DataSeries and **DataFrames** are built for easy manipulation and analysis of data. There are many commands in the Maple language that can be applied to these structures, including most Statistics commands. Many commands are also available from the right-click context menu. **DataSeries** and **DataFrames** also contain many commands, such as:

- Account for missing values using the FillMissing and DropMissing commands
- Find and remove duplicate entries using the AreDuplicate and AreUnique commands
- **DataFrames** are mutable; add rows or columns with Append
- Compute Aggregate statistics based on values in a column
- convert **DataSeries** and **DataFrames** to various other data storage types and change the datatype in place for **DataSeries**
- Subset and index into data using a natural labeled index or various Boolean queries

DataSeries

- A DataSeries is a one-dimensional sequence of data with a label for each data point. For example, you can keep track of nutritional energy values (in kJ per 100 g) of certain types of berries, as follows:

```
> energy := DataSeries(<220, 288, 136>, labels = [Raspberry,
  Grape, Strawberry]);
```

$$energy := \left[\begin{array}{cc} Raspberry & 220 \\ Grape & 288 \\ Strawberry & 136 \end{array}\right]$$

- This allows you to access the energy values by position (number) or label (name).

```
> energy[2];
```

$$288$$

```
> energy[Strawberry];
```

$$136$$

- You can determine which values satisfy some criteria by using elementwise operators. The result is a **DataSeries** of true/false values.

```
> energy >~ 200;
```

$$\left[\begin{array}{cc} \mathit{Raspberry} & \mathit{true} \\ \mathit{Grape} & \mathit{true} \\ \mathit{Strawberry} & \mathit{false} \end{array}\right]$$

- You can use this **DataSeries** to filter the entries in the original **DataSeries**.

```
> energy[energy >~ 200];
```

$$\left[\begin{array}{cc} \mathit{Raspberry} & 220 \\ \mathit{Grape} & 288 \end{array}\right]$$

DataFrame

- A DataFrame is a two-dimensional rectangular table of data with a label for each column and for each row. For example, you can keep track of various properties of certain types of berries as follows:

```
> genus := <"Rubus", "Vitis", "Fragaria">:
```

```
> carbohydrates := <11.94, 18.1, 7.68>:
```

```
> total_tons := < 543421, 58500118, 4594539 >:
```

```
> top_producer := < Russia, China, USA >:
```

```
> berry_data := DataFrame([genus, energy, carbohydrates,
  total_tons, top_producer], columns = [Genus, Energy,
  Carbohydrates, `Total tons`, `Top producer`], rows = Labels
  (energy));
```

$$\mathit{berry\_data} := \left[\begin{array}{cccccc} & \mathit{Genus} & \mathit{Energy} & \mathit{Carbohydrates} & \mathit{Total\ tons} & \mathit{Top\ producer} \\ \mathit{Raspberry} & \text{"Rubus"} & 220 & 11.94 & 543421 & \mathit{Russia} \\ \mathit{Grape} & \text{"Vitis"} & 288 & 18.1 & 58500118 & \mathit{China} \\ \mathit{Strawberry} & \text{"Fragaria"} & 136 & 7.68 & 4594539 & \mathit{USA} \end{array}\right]$$

Note that in the above example, the data stored in the **DataFrame** is heterogeneous;

each **DataSeries** has a different data type: **Float**, **Integer**, **string**, and **name**.

- You can access columns by indexing the berry **DataFrame** with a number, for the position, or a name. Each column is a **DataSeries**.

```
> berry_data[4];
```

$$\begin{bmatrix} \mathit{Raspberry} & 543421 \\ \mathit{Grape} & 58500118 \\ \mathit{Strawberry} & 4594539 \end{bmatrix}$$

```
> berry_data[Carbohydrates];
```

$$\begin{bmatrix} \mathit{Raspberry} & 11.94 \\ \mathit{Grape} & 18.1 \\ \mathit{Strawberry} & 7.68 \end{bmatrix}$$

- Because columns are **DataSeries**, you can test properties like for **DataSeries**.

```
> berry_data[Energy] >~ 200;
```

$$\begin{bmatrix} \mathit{Raspberry} & \mathit{true} \\ \mathit{Grape} & \mathit{true} \\ \mathit{Strawberry} & \mathit{false} \end{bmatrix}$$

- You can also filter rows. This returns a new **DataFrame** with a subset of the data.

```
> berry_data[berry_data[Energy] >~ 200];
```

$$\left[\begin{array}{cccccc} & \mathit{Genus} & \mathit{Energy} & \mathit{Carbohydrates} & \mathit{Total\ tons} & \mathit{Top\ producer} \\ \mathit{Raspberry} & \text{"Rubus"} & 220 & 11.94 & 543421 & \mathit{Russia} \\ \mathit{Grape} & \text{"Vitis"} & 288 & 18.1 & 58500118 & \mathit{China} \end{array}\right]$$

- By using the with command, you can simplify the syntax a little: the column names then represent the corresponding column directly, without the use of indexing.

```
> with(berry_data);
```

$$[\mathit{Genus}, \mathit{Energy}, \mathit{Carbohydrates}, \mathit{Total\ tons}, \mathit{Top\ producer}]$$

```
> Carbohydrates;
```

$$\begin{bmatrix} \mathit{Raspberry} & 11.94 \\ \mathit{Grape} & 18.1 \\ \mathit{Strawberry} & 7.68 \end{bmatrix}$$

```
> berry_data[Energy >~ 200];
```

$$\left[\begin{array}{cccccc} & \mathit{Genus} & \mathit{Energy} & \mathit{Carbohydrates} & \mathit{Total\ tons} & \mathit{Top\ producer} \\ \mathit{Raspberry} & \text{"Rubus"} & 220 & 11.94 & 543421 & \mathit{Russia} \\ \mathit{Grape} & \text{"Vitis"} & 288 & 18.1 & 58500118 & \mathit{China} \end{array}\right]$$