PUMA Repository

Pascal Units for Medical Applications

Unit *UnitConverter*

Lazarus and Free Pascal Edition

Version 1.1
Document version 1.1.4
2013-10-23

Johannes W. Dietrich

The Unit Converter is a Pascal unit that provides functions for parsing measurements consisting of measurement values and units. It also supports conversion of measurements from one unit to another one.

The Unit Converter provides the following data structures and functions:

tMeasurement

```
tMeasurement = record
  Value: extended;
  uom: string;
end;
```

tMeasurement is a record that holds both a numeric value (field *Value*) and a unit of measurement (field *uom*).

tUnitElements

```
tUnitElements = record
   MassPrefix, MassUnit, VolumePrefix, VolumeUnit: String;
end;
```

tUnitElements is a record that represents a structured form of a measurement unit.

Example: The unit “mg/dl” is represented as:
MassPrefix: “m”
MassUnit: “g”
VolumePrefix: “d”
VolumeUnit: “l”

tFloatFormat

```
TFloatFormat = (ffGeneral, ffExponent, ffFixed, ffNumber,
ffCurrency);
```

tFloatFormat is inherited from Free Pascal. See the official Lazarus and Free Pascal documentation for reference.

DecodeGreek

```
function DecodeGreek(theString: string): string;
```

DecodeGreek decodes an ASCII substitution sequence ("mc") for greek mu letter.

Example: “mcg/dl” is decoded as “µg/dl”.

EncodeGreek

```
function EncodeGreek(theString: string): string;
```

EncodeGreek encodes an ASCII substitution sequence for the greek mu letter.

Example: “µg/dl” is encoded as “mcg/dl”.

ParsedUnitString

```
function ParsedUnitString(theString: String):
TUnitElements;
```

ParsedUnitString parses a string for parts of a measurement unit and breaks it up into single components of a *TUnitElements* record.

ParsedMeasurement

```
function ParsedMeasurement(measurement: string):
tMeasurement;
```

ParsedMeasurement decomposes a measurement result into numeric value and unit.

Example: “2.5 mU/l” is decomposed to the numeric value 2.5 and the measurement unit “mU/l”.

ConvertedValue

```
function ConvertedValue(value, molarMass: real; fromUnit,
toUnit: string): real;
```

ConvertedValue converts a value delivered as a real number from one measurement unit to another one. Returns a real number.

Example: Triiodothyronine in a concentration of 5 pmol/l is converted to 3.2 (pg/ml) with *fromUnit* = “pmol/l”, *toUnit* = “pg/ml” and the molar mass of T3 (650.97 g/mol).

ValueFromUnit

```
function ValueFromUnit(fromValue: string; molarMass: real;
toUnit: string): real;
```

ValueFromUnit converts a value from one measurement unit to another one and delivers a numeric result.

Example: `ValueFromUnit('1.8 ng/dl'; 776.87; 'pmol/l')` delivers *23.1*.

UnitFromValue

```
function UnitFromValue(value, molarMass: real; fromUnit,
toUnit: string): string;
```

UnitFromValue converts a value in numeric form from one measurement unit to another one and delivers a string consisting of numeric value and measurement unit.

UnitFromValueF

```
function UnitFromValueF(value, molarMass: real; fromUnit,
toUnit: string; format: TFloatFormat; precision, digits:
integer): string;
```

UnitFromValueF converts a value in numeric form from one measurement unit to another one and delivers a formatted string consisting of numeric value and measurement unit.

ConvertedUnit

```
function ConvertedUnit(fromValue: string; molarMass: real;
toUnit: string): string;
```

ConvertedUnit converts a value from one measurement unit to another one and delivers a string with the result.

ConvertedUnitF

```
function ConvertedUnitF(fromValue: string; molarMass:
real; toUnit: string; format: TFloatFormat; precision,
digits: integer): string;
```

ConvertedUnit converts a value from one measurement unit to another one and delivers a formatted string with the result.

Example: `ValueFromUnit('18 ng/l'; 776.87; 'pmol/l', ffNumber, 2, 1)` delivers *“23.1 pmol/l”*.