Getting position data from TPositionInfoBase

Overview

This snippet demonstrates two ways of using `TPositionInfoBase`:

- getting the `TPosition` class from `TPositionInfoBase`
- showing `TPosition` in textual format

Note: The Location capability is included in the self-signing capabilities of S60 3rd Edition, Feature Pack 2 and newer platforms.

MMP file

The following libraries and capabilities are required:

```
CAPABILITY Location
LIBRARY lbs.lib
```

Header

```
#include <LbsPositionInfo.h>
#include <LbsPosition.h>

// Degrees sign delimeter used in formatting methods
_LIT(KDelimDegree,"\xb0"); // "°" symbol

// Dot delimeter used in formatting methods
_LIT(KDelimDot,"\x2e"); // "." symbol

// Plus sign delimeter used in formatting methods
_LIT(KDelimPlus,"\x2b"); // "+" symbol

// Minus sign delimeter used in formatting methods
_LIT(KDelimMinus,"\x2d"); // "-" symbol

// Quotation sign delimeter used in formatting methods
_LIT(KDelimQuot,"\x22"); // "\"" symbol

// Apostrophe sign delimeter used in formatting methods
_LIT(KApostrophe,"\x27"); // "'" symbol

const TInt KDegreeLength = 19;
```

Source

```
void CMyClass::PositionUpdated(TPositionInfoBase& aPosInfo)
    {
    // Check if position information class type is TPositionInfo
```


(C) Copyright Nokia 2014. All rights reserved.

```
    if (aPosInfo.PositionClassType() & EPositionInfoClass)
        {
        // Cast the TPositionInfoBase object to TPositionInfo
        TPositionInfo* posInfo = static_cast<TPositionInfo*>(&aPosInfo);

        // Get position
        TPosition position;
        posInfo->GetPosition(position);

        // Convert positions to the descriptors
        TBuf<KDegreeLength> latitudeDegr;
        GetDegreesString(position.Latitude(), latitudeDegr);

        // Convert positions to the descriptors
        TBuf<KDegreeLength> longitudeDegr;
        GetDegreesString(position.Longitude(), longitudeDegr);
        }
    }
```

```
void CMyClass::GetDegreesString(
    const TReal64& aDegrees,TBuf<KDegreeLength>& aDegreesString) const
    {
    const TReal KSecondsInMinute = 60.0;
    const TInt KNumWidth = 3;

    // If the aDegree is a proper number
    if ( !Math::IsNaN(aDegrees) )
        {
        // Integer part of the degrees
        TInt intDegrees = static_cast<TInt>(aDegrees);

        // Positive float of the degrees
        TReal64 realDegrees = aDegrees;

        // Convert to positive values
        if ( intDegrees < 0 )
            {
            intDegrees = -intDegrees;
            realDegrees = -realDegrees;
            }

        // Minutes
        TReal64 realMinutes = (realDegrees - intDegrees) * KSecondsInMinute;

        // Integer part of the minutes
        TInt intMinutes = static_cast<TInt>(realMinutes);

        // Seconds
        TReal64 realSeconds = (realMinutes - intMinutes) * KSecondsInMinute;
        TInt intSeconds = static_cast<TInt>((realMinutes - intMinutes) *
KSecondsInMinute);

        // Check the sign of the result
        if ( aDegrees >= 0 )
```


(C) Copyright Nokia 2014. All rights reserved.

```
            {
            aDegreesString.Append(KDelimPlus);
            }
        else
            {
            aDegreesString.Append(KDelimMinus);
            }

        // Add the degrees
        TInt64 value = intDegrees;
        aDegreesString.AppendNum(value);

        // Add the separator
        aDegreesString.Append(KDelimDegree);

        // Add the minutes
        value = intMinutes;
        aDegreesString.AppendNum(value);

        // Add the separator
        aDegreesString.Append(KApostrophe);

        // Add the seconds
        value = intSeconds;
        aDegreesString.AppendNum(value);

        // Add the separator
        aDegreesString.Append(KDelimQuot);

        // Add the separator
        aDegreesString.Append(KDelimDot);

        // Get six last digits
        realSeconds -= intSeconds;
        realSeconds *= 1000;

        // Add the seconds
        aDegreesString.AppendNumFixedWidth(static_cast<TInt>(realSeconds),
        EDecimal, KNumWidth);
        }
    }
```

Postconditions

Position data is read from `TPositionInfoBase`.

Version Hint

Windows Phone: [[Category:Windows Phone]]
[[Category:Windows Phone 7.5]]
[[Category:Windows Phone 8]]

Nokia Asha: [[Category:Nokia Asha]]
[[Category:Nokia Asha Platform 1.0]]

Series 40: [[Category:Series 40]]

(C) Copyright Nokia 2014. All rights reserved.

[[Category:Series 40 1st Edition]] [[Category:Series 40 2nd Edition]]
[[Category:Series 40 3rd Edition (initial release)]] [[Category:Series 40 3rd Edition FP1]] [[Category:Series 40 3rd Edition FP2]]
[[Category:Series 40 5th Edition (initial release)]] [[Category:Series 40 5th Edition FP1]]
[[Category:Series 40 6th Edition (initial release)]] [[Category:Series 40 6th Edition FP1]] [[Category:Series 40 Developer Platform 1.0]] [[Category:Series 40 Developer Platform 1.1]] [[Category:Series 40 Developer Platform 2.0]]

Symbian: [[Category:Symbian]]
[[Category:S60 1st Edition]] [[Category:S60 2nd Edition (initial release)]] [[Category:S60 2nd Edition FP1]] [[Category:S60 2nd Edition FP2]] [[Category:S60 2nd Edition FP3]]
[[Category:S60 3rd Edition (initial release)]] [[Category:S60 3rd Edition FP1]] [[Category:S60 3rd Edition FP2]]
[[Category:S60 5th Edition]]
[[Category:Symbian^3]] [[Category:Symbian Anna]] [[Category:Nokia Belle]]

See also

- Retrieving location information using Symbian C++
- S60 Platform Location Example

(C) Copyright Nokia 2014. All rights reserved.