Dynamic Properties

How to attach a Dynamic Property to CanderaObject

An application may want to attach any kind of application specific data to an arbitrary CanderaObject. This is easily possible as CanderaObject derives from DynamicPropertyHost, which can store dynamic properties. Different to conventional object properties, a dynamic property can be attached to an object, without being declared in the object itself. Further, a dynamic property does not allocate memory unless its value is set explicitly and differs from its default value.

Find below a simple example how to attach an application specific object, acting as dynamic property, to an object that derives from DynamicPropertyHost, like e.g. CanderaObject, Node, Mesh, Appearance, Texture, only to name a few.

- If possible the dynamic property definition (see class LayouterDynamicProperties) should be contained in the source file and not in the header file. Otherwise, the code size may get unnecessarily increased due to the inability of Gcc and Multi linker to get rid of duplicate template instantiation in the object files.
- Dynamic properties
  - can be set (see SetSize and SetLayoutDirection)
  - can be retrieved (see GetSize and GetLayoutDirection)
  - can be checked if they are set (see IsSizeSet and IsLayoutDirectionSet)
  - can be cleared (see ClearLayoutDirection)
  - have default values: C++ default initialization for simple types (see LayoutDirection) or specific values for complex types (see Size)
  - can have an optional change notification (see OnSizeChanged and OnLayoutDirectionChanged)
- A convenience interface should be provided to set, get, clear and check the dynamic property (see class Layouter). The dynamic property is only accessed via that interface.
- The DynamicPropertyHost (see class Layouter) has to provide the ParentProvider method for inherited dynamic properties.
  - In the Layouter sample it is guaranteed by the Layouter convenience interface that only CanderaObject instances are supported. Hence, a static_cast to CanderaObject is safe.

- Layouter.h

```
class Layouter : public CanderaObject
{
    …
    static void SetSize(CanderaObject& node, const Vector2& size);
```

```
    static const Vector2& GetSize(const CanderaObject& node);
    static bool IsSizeSet(const CanderaObject &node);
    …
    static void SetLayoutDirection(CanderaObject& node,
LayoutAlignment::LayoutDirection::Enum direction);
    static LayoutAlignment::LayoutDirection::Enum
 GetLayoutDirection(const CanderaObject& node);
    static bool IsLayoutDirectionSet(const CanderaObject &node);
    static void ClearLayoutDirection(CanderaObject& node);
    …
    static const Candera::DynamicProperties::DynamicPropertyHost
* ParentProvider(const Candera::DynamicProperties::DynamicPropertyHost* host);
    …
    static const Vector2& SizeDefault();
    …
    static void OnSizeChanged(DynamicPropertyHost* obj, const DynamicProperties::ValueChang
    …
    static void OnLayoutDirectionChanged(DynamicPropertyHost* obj, const DynamicProperties:
    …
};
```

- Layouter.cpp

```
class LayouterDynamicProperties
{
    …
    static void SetSize(CanderaObject& node, const Vector2& size)
    {
        static_cast<void>(node.SetValue(CdaDynamicPropertyInstance(Size), size));
    }

    static const Vector2& GetSize(const CanderaObject& node)
    {
        return node.GetValue(CdaDynamicPropertyInstance(Size));
    }

    static bool IsSizeSet(const CanderaObject &node)
    {
        return node.IsValueSet(CdaDynamicPropertyInstance(Size));
    }
    …
    static void SetLayoutDirection(CanderaObject& node,
LayoutAlignment::LayoutDirection::Enum direction)
    {
        static_cast<void>(node.SetValue(CdaDynamicPropertyInstance
(LayoutDirection), direction));
    }
```

```
    static LayoutAlignment::LayoutDirection::Enum
 GetLayoutDirection(const CanderaObject& node)
    {
        return node.GetValue(CdaDynamicPropertyInstance(LayoutDirection));
    }

    static bool IsLayoutDirectionSet(const CanderaObject& node)
    {
        return node.IsValueSet(CdaDynamicPropertyInstance(LayoutDirection));
    }

    static void ClearLayoutDirection(CanderaObject& node)
    {
        static_cast<void>(node.ClearValue(CdaDynamicPropertyInstance
(LayoutDirection)));
    }
    …
    CdaDynamicPropertiesDefinition(Candera::Layouter, CanderaObject);
        …
        CdaDynamicProperty(Size, Vector2);
            CdaDynamicPropertyValueChangedCb(&Layouter::OnSizeChanged);
            CdaDynamicPropertyDefaultValue(Layouter::SizeDefault());
            …
        CdaDynamicPropertyEnd();
        …
        CdaDynamicProperty(LayoutDirection, LayoutAlignment::LayoutDirection::Enum);
            CdaDynamicPropertyValueChangedCb(&Layouter::OnLayoutDirectionChanged);
            …
        CdaDynamicPropertyEnd();
        …
    CdaDynamicPropertiesEnd();
};
…
void Layouter::SetSize(CanderaObject& node, const Vector2& size)
{
    LayouterDynamicProperties::SetSize(node, size);
}

const Vector2& Layouter::GetSize(const CanderaObject& node)
{
    return LayouterDynamicProperties::GetSize(node);
}

bool Layouter::IsSizeSet(const CanderaObject &node)
{
    return LayouterDynamicProperties::IsSizeSet(node);
}
…
```

```
void Layouter::SetLayoutDirection(CanderaObject& node,
LayoutAlignment::LayoutDirection::Enum direction)
{
    LayouterDynamicProperties::SetLayoutDirection(node, direction);
}

LayoutAlignment::LayoutDirection::Enum
 Layouter::GetLayoutDirection(const CanderaObject& node)
{
    return LayouterDynamicProperties::GetLayoutDirection(node);
}

bool Layouter::IsLayoutDirectionSet(const CanderaObject& node)
{
    return LayouterDynamicProperties::IsLayoutDirectionSet(node);
}

void Layouter::ClearLayoutDirection(CanderaObject& node)
{
    LayouterDynamicProperties::ClearLayoutDirection(node);
}
…
const Candera::DynamicProperties::DynamicPropertyHost* Layouter::ParentProvider(const
Candera::DynamicProperties::DynamicPropertyHost* host) {
    CANDERA_SUPPRESS_LINT_FOR_NEXT_EXPRESSION(1774, "Layouter interface only allows Candera
    const CanderaObject* object = static_cast<const CanderaObject*>(host);
#ifdef CANDERA_2D_ENABLED
    const Node2D* node2D = Dynamic_Cast<const Node2D*>(object);
    if (0 != node2D) {
        return node2D->GetParent();
    }
#endif
#ifdef CANDERA_3D_ENABLED
    const Node* node = Dynamic_Cast<const Node*>(object);
    if (0 != node) {
        return node->GetParent();
    }
#endif
    return 0;
}
void Layouter::Layout(const AbstractNodePointer& node)
{
…
        Vector2 nodeSize(Layouter::GetSize(*node.ToCanderaObject()));
…
}
```

Revision #3
Created 2023-05-01 01:26:45 UTC by Tsuyoshi.Kato

Updated 2023-06-19 05:02:31 UTC by Tsuyoshi.Kato