OpenRTM-aist (C++) - 機能 #2333

coil::Factoryに生成したオブジェクトを管理する機能を追加する。

2012/01/31 00:28 - n-ando

ステータス:	終了	**開始日:**	2012/01/31
優先度:	通常	**期日:**	
担当者:	n-ando	**進捗率:**	100%
カテゴリ:		**予定工数:**	0.00時間
対象バージョン:			

説明

coil::Factoryに生成したオブジェクトを管理する機能を追加する。

```
        /*!
          *  @if  jp
          *
          *  @brief  生成済みオブジェクトリストの取得
          *
          *  このファクトリで生成されたオブジェクトのリストを取得する。
          *
          *  @return  生成済みオブジェクトリスト
          *
          *  @else
          *
          *  @brief  Getting  created  objects
          *
          *  This  operation  returns  a  list  of  created  objects  by  the  factory.
          *
          *  @return  created  object  list
          *
          *  @endif
          */
        std::vector<AbstractClass*>  createdObjects()
        {
            std::vector<AbstractClass*>  objects;
            for  (ObjectMapIt  it(m_objects.begin());  it  !=  m_objects.end();  ++it)
                {
                    objects.push_back(it->first);
                }
            return  objects;
        }

        /*!
          *  @if  jp
          *
          *  @brief  オブジェクトがこのファクトリの生成物かどうか調べる
          *
          *  @param  obj  対象オブジェクト
          *  @return  true:  このファクトリの生成物
          *                  false:  このファクトリの生成物ではない
          *
          *  @else
          *
          *  @brief  Whether  a  object  is  a  product  of  this  factory
          *
          *  @param  obj  A  target  object
          *  @return  true:  The  object  is  a  product  of  the  factory
          *                  false:  The  object  is  not  a  product  of  the  factory
          *
          *  @return  created  object  list
          *
          *  @endif
          */
```

```
    bool isProducerOf(AbstractClass* obj)
    {
        return m_objects.count(obj) != 0;
    }

    /*!
     * @if jp
     *
     * @brief オブジェクトからクラス識別子(ID)を取得する
     *
     * 当該オブジェクトのクラス識別子(ID)を取得する。
     *
     * @param obj [in] クラス識別子(ID)を取得したいオブジェクト
     * @param id [out] クラス識別子(ID)
     * @return リターンコード NOT_FOUND: 識別子が存在しない
     *                                    FACTORY_OK: 正常終了
     * @else
     *
     * @brief Getting class identifier (ID) from a object
     *
     * This operation returns a class identifier (ID) from a object.
     *
     * @param obj [in] An object to investigate its class ID.
     * @param id [out] Class identifier (ID)
     * @return Return code NOT_FOUND: ID not found
     *                                    FACTORY_OK: normal return
     * @endif
     */
    ReturnCode objectToIdentifier(AbstractClass* obj, Identifier& id)
    {
        if (m_objects.count(obj) == 0) { return NOT_FOUND; }
        id = m_objects[obj].id_;
        return FACTORY_OK;
    }

    /*!
     * @if jp
     *
     * @brief オブジェクトのコンストラクタを取得する
     *
     * このファクトリで生成されたオブジェクトのコンストラクタを取得する。
     * obj はこのファクトリで生成されたものでなければならない。予め
     * isProducerOf() 関数で当該オブジェクトがこのファクトリの生成物で
     * あるかどうかをチェックしなければならない。
     *
     * @return オブジェクトのデストラクタ
     *
     * @else
     *
     * @brief Getting destructor of the object
     *
     * This operation returns a constructor of the object created by
     * the factory.  obj must be a product of the factory.  User must
     * check if the object is a product of the factory by using
     * isProducerOf()-function, before using this function.
     *
     * @return destructor of the object
     *
     * @endif
     */
    Creator objectToCreator(AbstractClass* obj)
    {
        return m_objects[obj].creator_;
    }

    /*!
     * @if jp
```

```
     *
     *  @brief  オブジェクトのデストラクタを取得する
     *
     *  このファクトリで生成されたオブジェクトのデストラクタを取得する。
     *  obj  はこのファクトリで生成されたものでなければならない。予め
     *  isProducerOf()  関数で当該オブジェクトがこのファクトリの生成物で
     *  あるかどうかをチェックしなければならない。
     *
     *  @return  オブジェクトのデストラクタ
     *
     *  @else
     *
     *  @brief  Getting  destructor  of  the  object
     *
     *  This  operation  returns  a  destructor  of  the  object  created  by
     *  the  factory.    obj  must  be  a  product  of  the  factory.    User  must
     *  check  if  the  object  is  a  product  of  the  factory  by  using
     *  isProducerOf()-function,  before  using  this  function.
     *
     *  @return  destructor  of  the  object
     *
     *  @endif
     */
    Destructor  objectToDestructor(AbstractClass*  obj)
    {
        return  m_objects[obj].destructor_;
    }
```

関連するチケット:

関連している OpenRTM-aist (Java) - 機能 #2331: coil::Factoryに生成した...	終了	2012/01/31

履歴

#1 - 2012/01/31 00:28 - n-ando

- ステータス を 新規 から 終了 に変更

- 進捗率 を 0 から 100 に変更