QAxWidget Class

QAxWidget 类是QWidget包装 ActiveX 控件。更多的...

Header:	**#include**
CMake:	find_package(Qt6 REQUIRED COMPONENTS AxContainer) target_link_libraries(mytarget PRIVATE Qt6::AxContainer)
qmake:	QT += axcontainer
Inherits:	QAxBaseWidget and QAxBase

- 所有成员的列表，包括继承的成员

公共职能

	QAxWidget(QWidget **parent* = nullptr, Qt::WindowFlags *f* = Qt::WindowFlags())
	QAxWidget(const QString &*c*, QWidget **parent* = nullptr, Qt::WindowFlags *f* = Qt::WindowFlags())
	QAxWidget(IUnknown **iface*, QWidget** parent* = nullptr, Qt::WindowFlags *f* = Qt::WindowFlags())
virtual	~QAxWidget() override
void	clear()
virtual QAxAggregated *	createAggregate()
bool	doVerb(const QString &*verb*)

重载的公共职能

virtual QSize	**minimumSizeHint() const override**
virtual void	resetControl() override
virtual QSize	sizeHint() const override

protected function

virtual bool	**createHostWindow**(**bool *initialized***)
bool	**createHostWindow**(bool *initialized*, const QByteArray &*data*)
virtual bool	**translateKeyEvent**(int *message*, int *keycode*) const

重载的protected function

virtual void	**changeEvent**(**QEvent *e**) **override**
virtual void	**connectNotify**(const QMetaMethod &*signal*) override
virtual bool	**initialize**(IUnknown **_ptr_) override
virtual void	**resizeEvent**(QResizeEvent *) override

详细说明

QAxWidget 可以实例化为空对象，并使用它应包装的 ActiveX 控件的名称，或者使用指向 ActiveX 控件的现有接口指针。ActiveX控件的属性、方法和事件仅使用QAxBase支持的数据类型，可作为 Qt 属性、槽和信号使用。基类QAxBase提供API直接通过指针访问ActiveX `IUnknown`。

QAxWidget 是一个QWidget并且主要可以这样使用，例如，它可以组织在小部件层次结构和布局中，或者充当事件过滤器。标准小部件属性，例如enabled支持，但它依赖于 ActiveX 控件来实现对环境属性（例如调色板或字体）的支持。QAxWidget 尝试提供必要的提示。

但是，您不能重新实现特定于 Qt 的事件处理程序（例如 mousePressEvent 或 keyPressEvent）并期望它们能够可靠地调用。嵌入式控件完全覆盖 QAxWidget，并且通常处理用户界面本身。使用特定于控件的 API（即监听控件的信号），或使用标准 COM 技术，如窗口过程子类化。

QAxWidget 还继承了大部分与 ActiveX 相关的功能QAxBase， 尤其dynamicCall（） 和querySubObject()。

警告：您可以子类化 QAxWidget，但不能`Q_OBJECT`在子类中使用宏（生成的 moc 文件将无法编译），因此您无法添加更多信号、槽或属性。此限制是由于运行时生成的元对象信息造成的。要解决此问题，请将 QAxWidget 聚合为QObject子类。

也可以看看QAxBase,QAxObject,QAxScript， 和ActiveQt Framework。

成员函数文档

*[explicit]QAxWidget::QAxWidget(QWidget *parent = nullptr, Qt::WindowFlags f = Qt::WindowFlags())*

创建一个空的 QAxWidget 小部件并传播*parent*和*f*到QWidget构造函数。要初始化控件，请调用setControl()。

*[explicit]QAxWidget::QAxWidget(const QString &c, QWidget *parent = nullptr, Qt::WindowFlags f = Qt::WindowFlags())*

创建 QAxWidget 小部件并初始化 ActiveX 控件*c*。*parent*和*f*被传播到QWidget构造函数。

也可以看看setControl()。

*[explicit]QAxWidget::QAxWidget(IUnknown **iface***, **QWidget** parent* = nullptr, Qt::WindowFlags f = Qt::WindowFlags())*

创建一个 QAxWidget，它包装了引用的 COM 对象*iface*。*parent*和*f*被传播到QWidget构造函数。

[override virtual]QAxWidget::~QAxWidget()

关闭 ActiveX 控件并销毁QAxWidget小部件，清理所有分配的资源。

也可以看看clear()。

*[override virtual protected]void QAxWidget::changeEvent(QEvent *e)*

重新实现：QWidget::changeEvent（QEvent *事件）。

void QAxWidget::clear()

关闭 ActiveX 控件。

也可以看看resetControl()。

[override virtual protected]void QAxWidget::connectNotify(const QMetaMethod &signal)

重新实现：QObject::connectNotify（常量 QMetaMethod 和信号）。

*[virtual]QAxAggregated *QAxWidget::createAggregate()*

当您想要为 ActiveX 控件的客户端站点实现其他 COM 接口，或者当您想要提供 COM 接口的替代实现时，请重新实现此函数。返回 a 的新对象QAxAggregated子类。

默认实现返回空指针。

[virtual protected]bool QAxWidget::createHostWindow(bool initialized)

为 ActiveX 控件创建客户端站点，如果该控件可以成功嵌入，则返回 true，否则返回 false。如果*initialized*为 true 时，控件已被初始化。

这个函数被调用initialize()。如果您重新实现初始化以自定义实际的控件实例化，请在重新实现中调用此函数以使默认客户端嵌入控件。为 ActiveX 控件创建客户端站点，如果该控件可以成功嵌入，则返回 true，否则返回 false。

[protected]bool QAxWidget::createHostWindow(bool initialized, const QByteArray &data)

为 ActiveX 控件创建客户端站点，如果该控件可以成功嵌入，则返回 true，否则返回 false。如果*initialized*为 false 时，控件将使用*data*。该控件将通过 IPersistStreamInit 或 IPersistStorage 接口进行初始化。

如果需要使用自定义数据初始化控件，请在重新实现中调用此函数initialize()。默认实现不调用此函数initialize()。

bool QAxWidget::doVerb(const QString &verb)

请求 ActiveX 控件执行操作*verb*。可能的动词由以下命令返回verbs()。

如果对象可以执行该操作，则该函数返回 true，否则返回 false。

*[override virtual protected]bool QAxWidget::initialize(IUnknown **ptr)*

重新实现：QAxBase::initialize（未知 **ptr）。

通话QAxBase::initialize（*ptr*），并通过调用将控件嵌入到该小部件中createHostWindow（假）如果成功。

要在激活控件之前对其进行初始化，请重新实现此函数并在调用之前添加初始化代码createHostWindow（真的）。

成功则返回`true`，`false`否则返回。

[override virtual]QSize QAxWidget::minimumSizeHint() const

重新实现属性的访问函数：QWidget::minimumSizeHint。

[override virtual]void QAxWidget::resetControl()

重新实现：QAxObjectInterface::resetControl()。

关闭 ActiveX 控件。

*[override virtual protected]void QAxWidget::resizeEvent(QResizeEvent *)*

重新实现：QWidget::resizeEvent（QResizeEvent *事件）。

[override virtual]QSize QAxWidget::sizeHint() const

重新实现属性的访问函数：QWidget::sizeHint。

[virtual protected]bool QAxWidget::translateKeyEvent(int message, int keycode) const

重新实现此函数以将某些按键事件传递给 ActiveX 控件。*message*是指定消息类型的 Window 消息标识符（即 WM_KEYDOWN），并且*keycode*是虚拟键码（即VK_TAB）。

如果函数返回 true，则按键事件将传递到 ActiveX 控件，然后 ActiveX 控件会处理该事件或将事件传递到 Qt。

如果函数返回 false，则 ActiveQt 会忽略按键事件的处理，即。ActiveX 控件可能会处理它，也可能不会处理它。

对于以下情况，默认实现返回 true：

WM_SYSKEYDOWN	WM_SYSKEYUP	WM_KEYDOWN
所有键码	VK_菜单	VK_TAB、VK_DELETE 以及所有非箭头键与 VK_SHIFT、VK_CONTROL 或 VK_MENU 的组合

该表是对流行的 ActiveX 控件进行实验的结果，即。Internet Explorer 和 Microsoft Office 应用程序，但对于某些控件可能需要修改。