Proje Adresi : https://goo.gl/Z4mmxS

Hazırlayanlar :

Velican Akkuş

Anıl Mert Kınay

Mustafa Akbay

Kaynaklar :

www.kaanaslan.com

plepa.com

Ödev Konusu : Noktasal değerler içeren bir grafikte seçilen bir noktaya en yakın komşuları belirleyerek sınıflandırma yapan bir program.

Ödev Amacı: Bir yapay zeka algoritmasını C++ kullanarak Gui araçları ile implemente etmek.

Kullanılan Kütüphane ve Framework Araçları : STL, QT

Qt Nedir?

Qt (genellikle "kyu:t" biçiminde okunuyor fakat "kyu ti" biçiminde de okunabiliyor) C++ tabanlı bir GUI ortamıdır (GUI framework). Bu ortamın ana amacı C++ kullanarak pencereli (yani GUI) uygulamalar geliştirmektir. Her ne kadar Qt'nin ana amacı GUI uygulamalar geliştirmek olsa da bu ortam zamanla pek çok gereksinimi karşılayacak kütüphanelere ve araçlara sahip duruma gelmiştir.

Anahtar Notlar: İngilizce "framework" sözcüğünü Türkçe "ortam" olarak kullanacağız. Örneğin "Qt Framework" yerine "Qt Ortamı" gibi. GUI ise "Graphical User Interface" sözcüklerinden kısaltılmıştır. "Grafik Kullanıcı Arayüzü" anlamına gelmektedir.

Qt'nin Kısa Tarihi

Qt ortamı bir kütüphane biçiminde 1990'lı yılların başlarında geliştirilmeye başlanmıştır. Geliştiriciler daha sonra Troll Tech isimli firmayı kurdular ve Qt bu firma tarafından geliştirilmeye devam etti. Qt'nin ilk versiyonları oldukça sade idi. Versiyonlar ilerledikçe kütüphaneye yeni özellikler eklendi. Qt'nin ilk versiyonları yalnızca UNIX/Linux sistemlerinde (yani X11 üzerinde) çalışıyordu. Daha sonra Qt diğer platformlarda da kullanılabilir yani "cross platform" hele getirildi. Böylece önce Windows sistemlerinde sonra da Mac OS X sistemlerinde Qt kullanılmaya başlandı.

Qt 4.0'la ve 5.0'la birlikte bazı önemli değişiklikler ve yenilikleri de bünyesine katmıştır. Bugün için Qt'nin son stabil versiyonu 5.10'dır. Qt'nin son zamanlardaki en önemli yenilikleri "Qt Quick" denilen ve QML sentaksıyla arayüz oluşturmaya izin veren platformudur. Aynı zamanda Qt Android ve IOS sistemlerinde de kullanılabilir hale getirilmiştir.

Qt'nin Kullanım Lisansları

Bugün Qt ticari (commercial), GPL, LGPL ve diğer bazı open source lisanslarla sunulmaktadır. Yani özetle bugün için hukuki durum şöyledir: Biz Qt'yi indirip kendi projelelerimizde istediğimiz gibi kullanabiliriz. Projemizi açmak ya da bedava dağıtmak zorunda değiliz. Ancak eğer Qt'nin kaynak kodları üzerinde değişiklik yapıp onu da açmak istemiyorsak Digia'ya para ödeyip ticari lisansa sahip olmamız gerekir.

Qt'de Diyalog Pencereleri

Her zaman üst penceresinin yukarısında görüntülenen "sahiplenilmiş (owned)" pencerelere diyalog pencereleri denilmektedir. Diyalog pencereleri kendi aralarında "modal" ve "modeless" olmak üzere ikiye ayrılmaktadır. Modal diyalog pencereleri açıldığında artık kapanana kadar onun üst penceresi olan arka planla etkileşim kalmaz. Örneğin MessageBox pencereleri tipik olarak modal pencerelerdir. Halbuki modeless pencerelerde arka plan etkileşimi devam etmektedir. Örneğin "Find and Replace" pencereleri tipik olarak "modeless" diyalog pencereleridir.

Modal Diyalog Pencerelerinin Oluşturulması

Modal diyalog pencereleri şu aşamalardan geçilerek oluşturulur:

1) Diyalog pencerelerinin işlevsellikleri QDialog sınıfıyla sağlanmaktadır. Dolayısıyla öncelikle QDialog sınıfından bir sınıfın türetilmesi gerekir. Aslında bu türetmenin elle yapılmasına gerek yoktur. Qt Creator'da proje üzerine gelinip bağlam menüsünden "Add New" seçilip çıkan diyalog penceresinden "Qt/Qt Designer Form Class" ile bu işlem otomatik olarak yapılabilir. Add New diyalog penceresinde ilerlerken seçenek olarak "Dialog With Buttons" seçilmelidir.

2) Diyalog penceresi açılacağı zaman ilgili diyalog sınıfından bir nesne yaratılır ve bu nesneyle QDialog sınıfından gelen exec fonksiyonu çağrılır. Modal diyalog nesnesinin yerel olmasında bir sakınca yoktur.

```
void MainWindow::on_m_actionSettings_triggered()
{
    SettingDialog sDialog(this, m_newPointSize, m_pointSize, m_lineSize,
m_node);
    sDialog.exec();
}
```

3) Diyalog penceresinin pencere başlığı QDialog sınıfının windowTitle fonksiyonuyla değiştirilebilir. Diyalog penceresini kapatmak için QDialog sınıfının done isimli fonksiyonu kullanılır. Tipik olarak bir modal diyalog penceresinde en azından Ok ve Cancel biçiminde iki düğme bulunur. İşte bu düğmelere tıklandığında done fonksiyonu çağrılmalıdır. done fonksiyuonun parametresi ddialog penceresinin kapanma nedenini belirtir. Tipik olarak QDialog::Accepted ve QDialog::Rejected değerleri tercih edilmektedir. Bu değerler exec fonksiyonun geri dönüş değeri biçiminde elde edilmektedir. Tipik olarak diyalog penceresini açan programcı exec fonksiyonunun geri dönüş değerini kontrol etmektedir.

```
void MainWindow::on_m_actionSettings_triggered()
{
    SettingDialog sDialog(this, m_newPointSize, m_pointSize, m_lineSize,
m_node);
    if (sDialog.exec() == QDialog::Accepted)
        //...
}
```

4) Pekiyi diyalog penceresinde oluşan bilgiler nasıl kullanılacaktır? Biz bu bilgileri doğrudan diyalog sınıfının ieçrisinden kullanabailiriz. Fakat daha çok bu bilgiler diyalog penceresini açan yerden kullanılmaktadır. done işlemi yapıldığında diyalog penceresinin içerisinde oluşan data'ların sınıfın veri elemanlarında saklanması uygun olur. Eğer bu veri elemanları private bölüme konulacaksa bunlar için get ve set fonksiyonlarının yazılması gerekir.

```
void MainWindow::on_m_actionSettings_triggered()
{
    SettingDialog sDialog(this, m_newPointSize, m_pointSize, m_lineSize,
m_node);
    if (sDialog.exec() == QDialog::Accepted)
        sDialog.XXX();
}
```

Qt'de Çizim İşlemleri

Windows gibi bazı sistemler pencere içerisindeki görüntüyü bizim için tutmamaktadır. Pencere içerisindeki görüntünün oluşturulması Windows sistemlerinde programcıya bırakılmıştır. Programcı pencere içerisine bir çizim yaptığında o pencerenin üzerine bir pencere getirilip tekrar açıldığında o çizim kaybolur. Windows bunu tutarak geri basmamaktadır. Bunun yerine WM_PAINT isimli mesajı thread'in mesaj kuyruğuna bırakır. Bu mesaj pencere içerisindeki görüntünün bozulduğu ve yeniden çizilmesi gerektiği anlamına gelmektedir. Dolayısıyla bu sistemlerde çizimler bu mesaj geldiğinde yapılmalıdır. Diğer işletim sistemlerini bazıları pencere içerisindeki çizimleri kendisi tutup basabilmektedir. Ancak Qt'nin "cross platform" özelliği ortak bir bölene göre tsarlanmıştır. Qt'de pencere içerisindeki çizim bozulduğunda framework tarafından QWidget sınıfının paintEvent isimli sanal fonksiyonu çağrılır. O halde çizimler bu fonksiyon içerisinde yapılmalıdır.

QPainter sınıfının çizim işlemlerini yapan pek çok drawXXX fonksiyonu vardır. drawXXX fonksiyonları çizimi yaparken bazı çizim nesnelerini kullanmaktadır. İki önemli çizim nesnesi vardır: Kalem (pen) ve fırça (brush). Kalem nesnesi QPen sınıfı ile fırça nesnesi de QBrush sınıfı ile temsil edilmiştir. Bir kalem ve fırça nesnesi yaratılıp QPainter sınıfının setPen ve setBrush fonksiyonlarıyla kullanıma hazır hale getirilebilir. Çizim işlemlerinde orijin noktası çalışma sol-üst köşesidir. X değeri sağa doğru Y değeri aşağıya doğru artar.

```
void MainWindow::paintEvent(QPaintEvent *)
{
        QPainter painter(this);
        QPen myPen(Qt::black);
        myPen.setWidth(5);
        painter.setPen(myPen);
        painter.drawPoint(QPoint (5, 5));
}
```