QObject (chap 8)

- a parent
- children
- a name
- no copy constructor
- no assignment operator

TWO QObject with same name

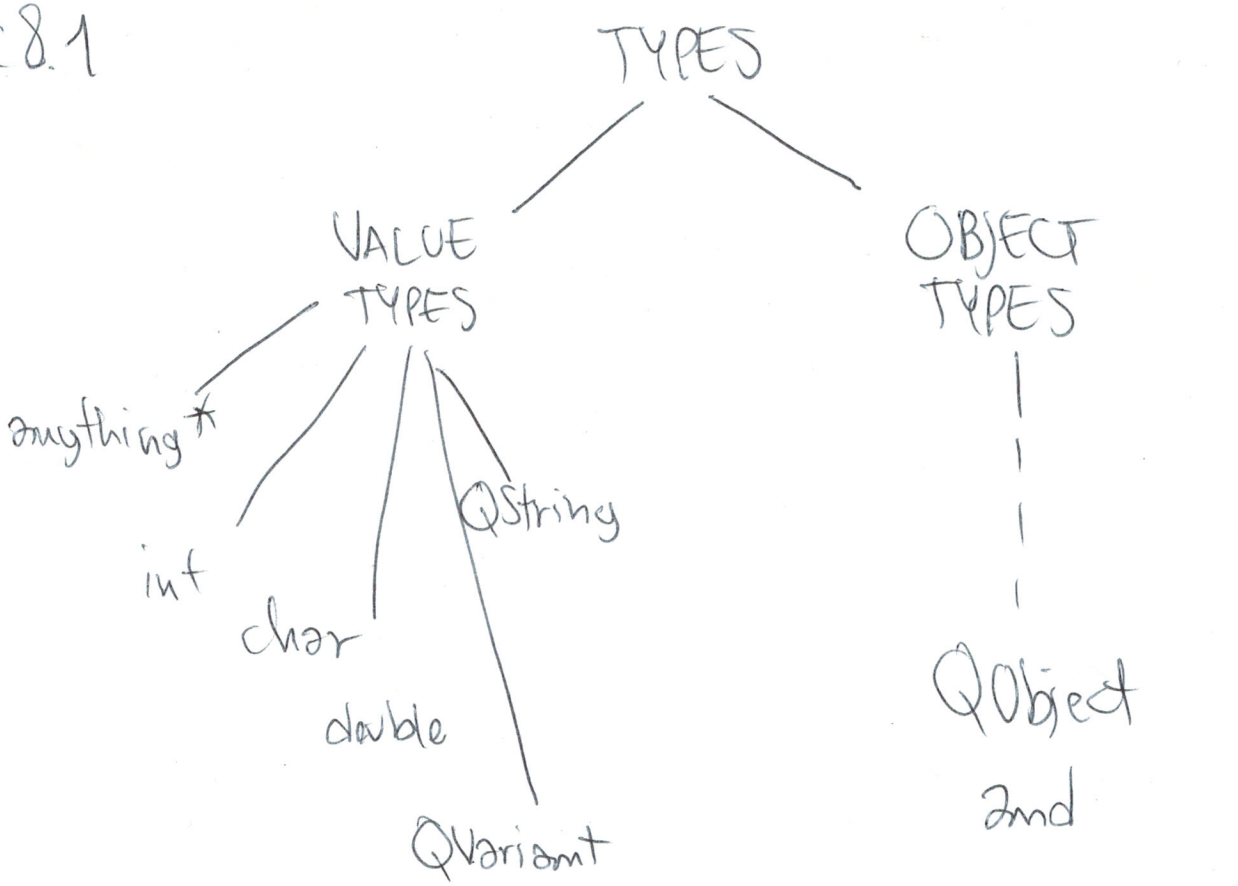
CANNOT EXIST

↓

EVERY QObject is UNIQUE

```
class QObject{
public:
    explict QObject(QObject* parent=0);
    QObject *parent() const;
    QString objectName();
    void setParent(QObject* parent);
```

SEC 8.1



ASSIGNABLE DATA TYPES

UNIQUE

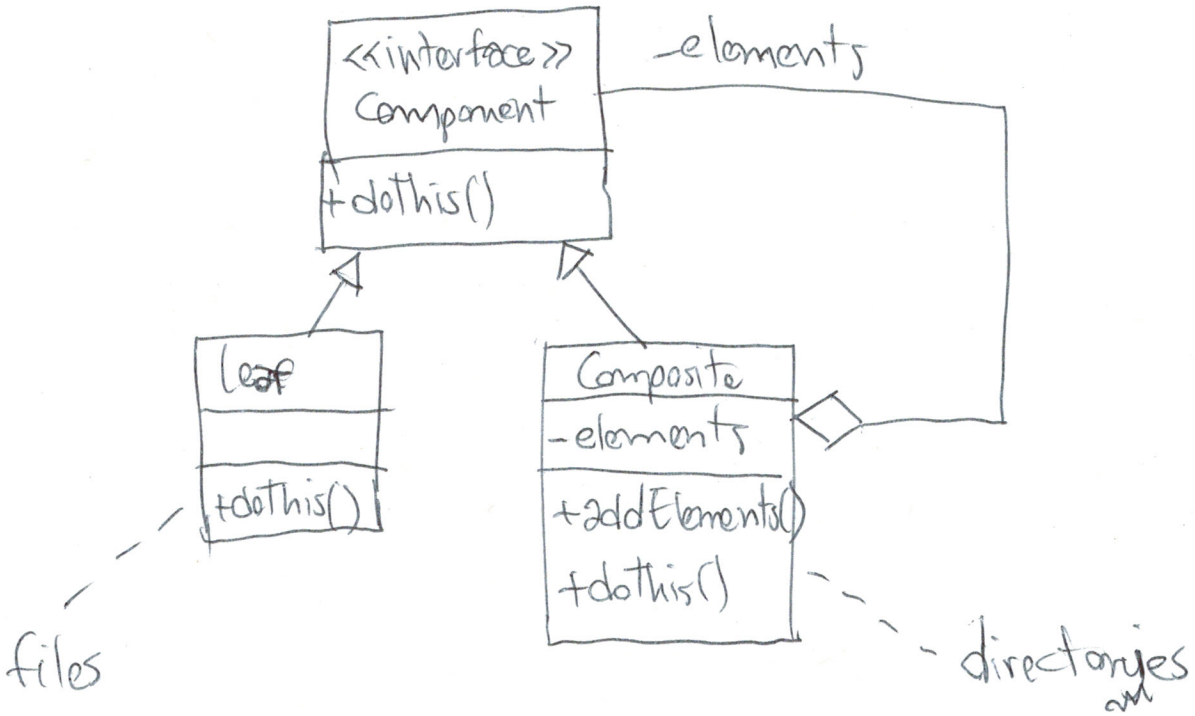
SEC 8.2 DESIGN PATTERNS

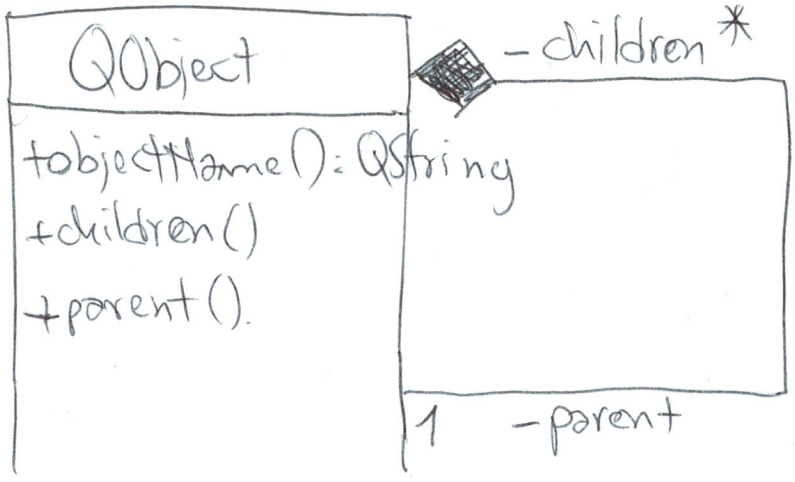
↓

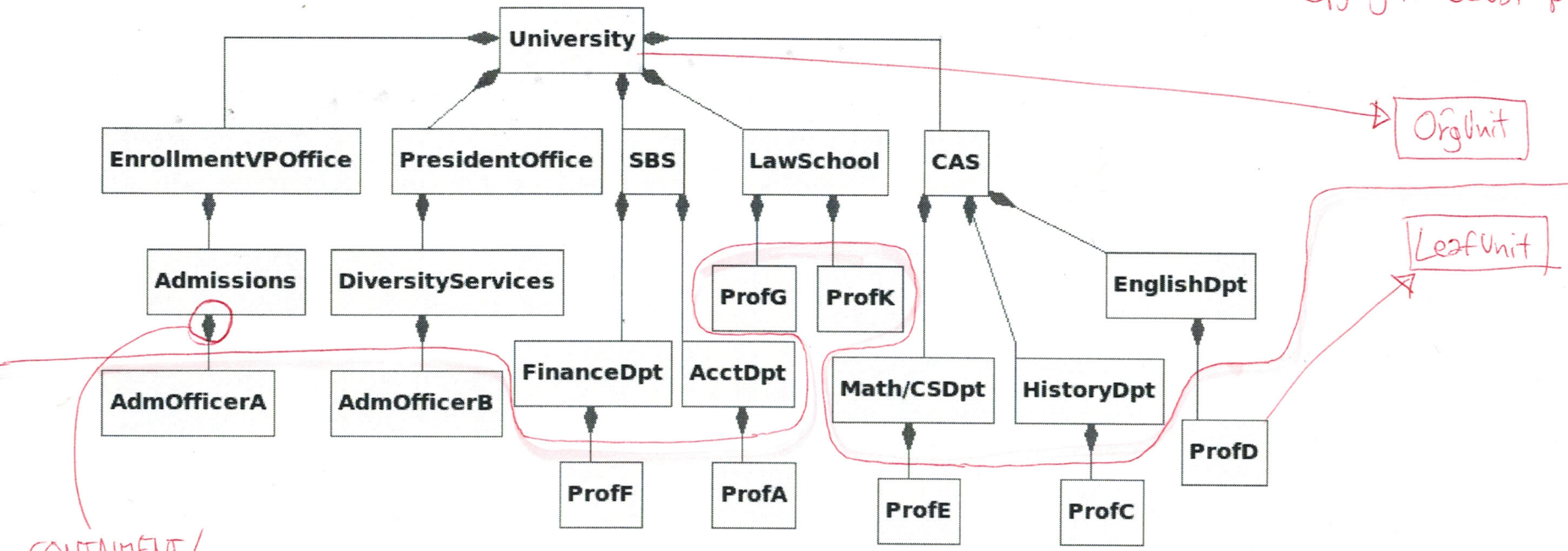
COMPOSITE PATTERN

EXAMPLE: dir / files

parent / children relationships




QObject
-children *
+objectName(): QString
+children()
+parent()
1
-parent



CONTAINMENT/ COMPOSITION

EACH BOX IS A COMPONENT

EACH NODE CAN BE:

```
class OrgUnit: public QObject{
public:
        QString getName();
        double getSalary();
private:
        QString m_Name;
        double  m_salary;
};
```

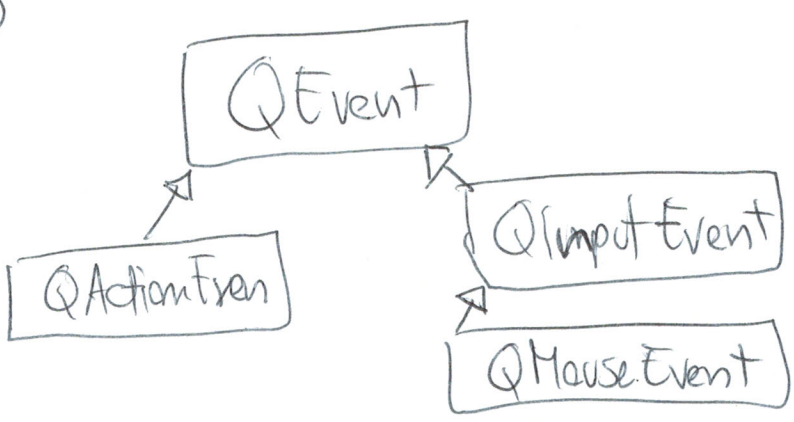
p 266

SEC 8.2.1 children()

QObject::children() — LOCAL CHILDREN

QList<T> parentObj.findChildren<T>
(const QString& name)

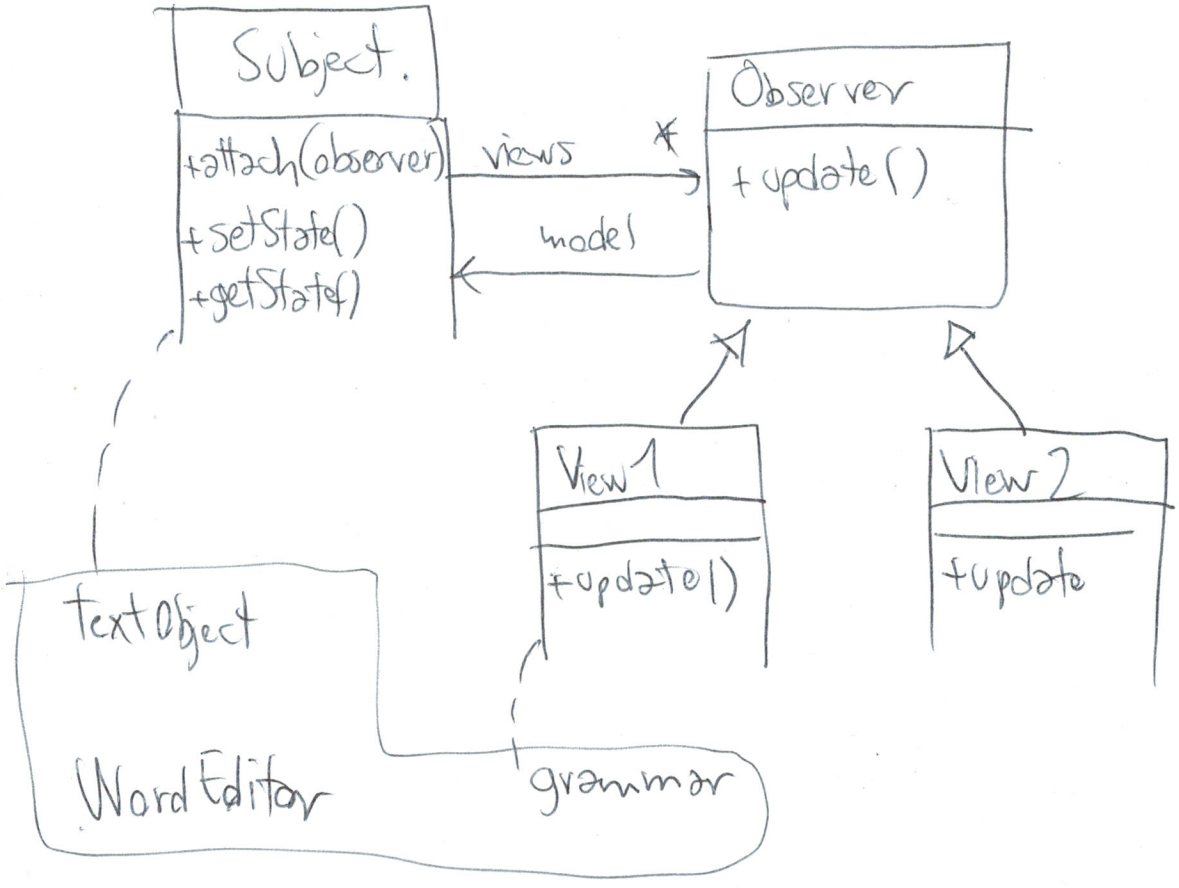
SEC 8.3



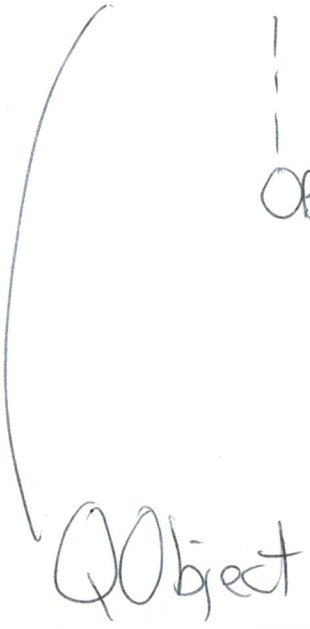
EVENT LOOP

OBSERVER (p272)

BEHAVIORAL



SEC 8.5 SIGNAL SLOTS

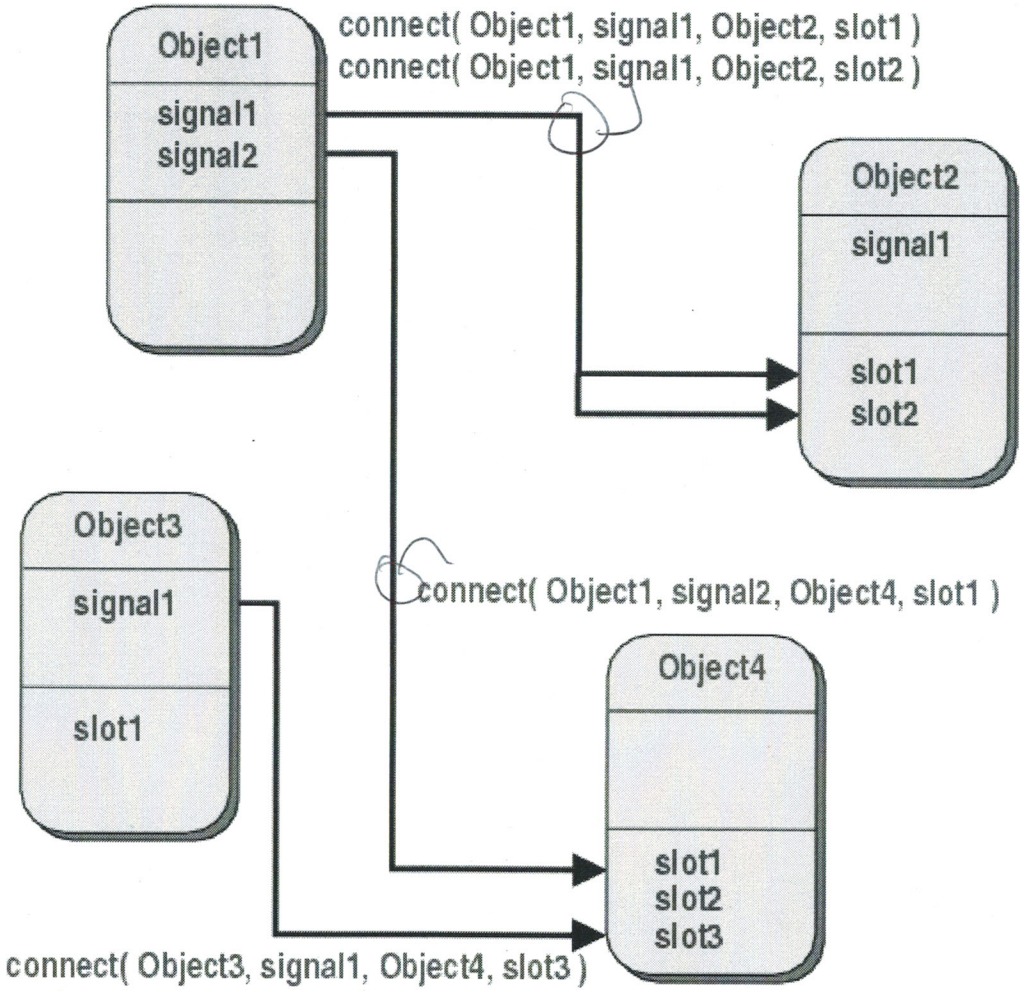


```
bool  QObject :: connect (
                    sender  QObject Ptr,
                    SIGNAL (signal Name(arguments)),
                    receiver QObject Ptr,
                    SLOT (slotName (arguments)),
                    connectionType);
```

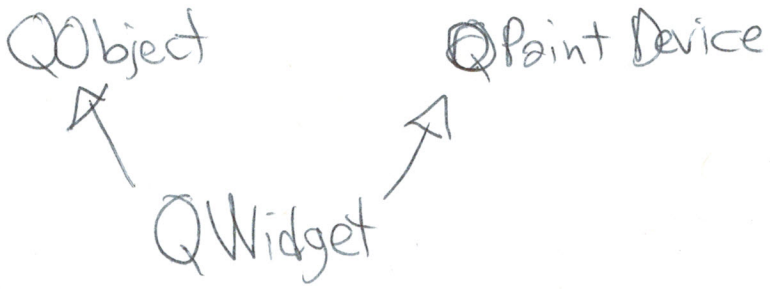
SEC 8.6 BEST PRACTICES:

create QApplication FIRST
and then all your QObject

WARN!
MUST DO



CHAP 9



WIDGET WITH NO PARENT ⟶ WINDOW