1.18. Review Questions

What is a stream? What kinds of streams are there?

A stream is an object which you can use for input or output. It accepts other objects via the insertion or extraction operators (<< and >>). A stream can be attached to a file. An input stream can be attached to the keyboard (such as standard input, by default). An output stream can be attached to the program console (as standard output and standard error are, by default). A stream can also be attached to a string.

Give one reason to use an ostrstream.

So that you can format and write to strings in memory, without sending anything to standard output/error.

What is the main difference between getline and the >> operator?

both operations skip leading whitespace, but the >> operator will stop reading when it encounters the first whitespace, while getline gets an entire line of input.

What is the type of each expression in the following list?

3.14

double

'D'

char

"d"

const char*

6

int

6.2f

float

"something stringy"

const char*

false

bool

In **Example 1.34**, identify the type and value of each numbered item:

Example 1.34. src/types/types.cpp

```
#include <QTextStream>


int main() {
        QTextStream cout(stdout);
        int i = 5;
        int j=6;
        int* p = &i;     ❶

        int& r=i;
        int& rpr=(*p);
        i = 10;
        p = &j;          ❷
        rpr = 7;         ❸

        r = 8;           ❹
        cout << "i=" << i << " j=" << j << endl;  ❺
        return 0;
}
```

❶ *p: 5

❷ *p: 6

❸ *p: 6

❹ rpr: 8

What is the difference between a pointer and a reference?

A pointer is a variable that stores the *address* of another variable. It must be dereferenced before it can access the value held by the other variable.

A reference is an *alias* for another variable. It can be used the same way that the variable can (unless the reference was declared to be const).

What is the keyword "const" used for? Why and how would you use it in a program?

What is the "address-of" operator? Why and how would you use it in a program?

the & and gives you the address of a variable

What is the "dereference" operator? Why and how would you use it in a program?
it is the asterisk *. It gives you value to which you are pointing to.
What is a null pointer? Why would you define one in a program?
a Null pointer is a pointer to any data type, but the value is 0. You would want to define one to help find bugs and is a good way to program to test.
What is a memory leak? What would cause one in a program?
What might cause a segmentation fault (or, in Windows, a general protection fault)?
Usuaully if you have a pointer pointing to something in memory that isn't mapped then it could cause it.
What are the possible uses of const when dealing with pointers?
What is a function's signature?
What is meant by the term, "function overloading" ?
Why is it an error to have two functions with the same signature but different return types in one scope?
- Because it is not allowed in C++ (C++11).
Why does main(int argc, char* argv[]) sometimes have parameters? What are they used for?

They are the arguments which were passed in from the command-line when running this program. argc is the number of command-line arguments, argv is an array of the actual arguments as char arrays (C-style strings).

2.16. Review Questions

Describe at least one difference between a class and a struct.

Basically, they are the same except for the default access level of members (public for struct, private for class). Classes and structs can both have member functions, constructors, etc. However, as soon as you add member functions and access specifiers, a struct becomes more "class-like".

Struct and class is equivalent but in struct, all members are public. But in class, you have public, private, and protected members which is not possible in a struct.
How does class scope differ from block scope?

Class scope is accessible inside other class members. block scope is accessible only inside that block.
Describe two situations where it is OK to use friend functions.

When enforcing creational rules, or for global operator functions.
Friend functions are functions that are able to access private and publicmember of the class.
How does a static data member differ from a non-static data member?

A static data member of a class T is created and initialized when the program begins execution. It exists even if no T objects have been created. It is shared by all existing T objects. It is destroyed when the program terminates.

A non-static data member of a class T is created and initialized only when a T object is created. It belongs exclusively to one object. It is destroyed when that object is destroyed.

What is the difference between a static member function and a non-static member function?

A static member function of a class T cannot access any non-static data members of T. If it is public, it can be accessed by client code with the scope resolution operator T::, or through an object/pointer/reference.

A non-static member function of a class T can access any data members of T. If it is public, it can be accessed by client code through an object (directly, or indirectly via pointer/reference).

What does it mean to declare a member function to be const?

The member function is not allowed to change any member of *this.
means that function does not change any data member of our clalss.
Explain what would happen (and why) if a class T could have a copy constructor with the following prototype?
`T::T(T other);`

This would be a compile error. If the compiler allowed that version of the copy constructor, then imagine what would happen during a pass-by-value. Because this copy constructor takes a value parameter, it would need to call itself to pass its own parameter, and this would result in an infinite recursion.

3.5. Review Questions

What is QTextStream and how can you reuse it?

The QTextStream class provides a convenient interface for reading and writing text. So far we have used QTextStreams to move data to or from a text file, to standard out (the screen), from standard in (the keyboard) and to a QString.

What is the purpose of the QT variable in the qmake project file? What are possible values?

It specifies which Qt modules will be enabled for this project. Possible values are gui, xml, sql, and net. core is enabled by default.

Why should you use QTextStream instead of iostream in your programs?

<QTextStream> works with Unicode QStrings and other Qt types. iostream does not work with any Qt types.

4.5. Review Questions

[fromfile: lists-questions.xml id: lists-questions]

Name three applications in which QStringList would be useful.

How can you convert a QStringList to a QString? Why might you want to do that?

How can you convert a QString to a QStringList? Why might you want to do that?

What is the Iterator pattern? Give three examples.

An object or programming structure that provides indirect access to each element in a container, without being bound to a specific underlying type.

the iterator pattern is the first pattern we discussed. With it, we are able to , in a standard way, walk through a data structor. Is very powerful as with an iterator, you don't need to know which data structure your class implements.

Examples: STL-style iterators, Java-styleIterators, and foreach loops.

Draw a UML diagram with three or more classes and make sure that all these different kinds of relationships are represented.

Aggregation and composition

One-to-One and One-to-Many

Unidirectional and bidirectional
The classes should represent real-world concepts, and the relationships should attempt to represent reality. Write a couple of paragraphs explaining why there is a relationship of each kind in your diagram.
What is the difference between composition and aggregation?

In a composition relationship one object (the one closest to the filled diamond) manages the object(s) at the other end (called components). The manager has responsibility for destroying the component(s)as part of its own destruction. In an aggregate relationship, the lifetimes of the objects on either end are not related to one another.
the difference between composition and aggregation. Composite relation will manage the data over lifecycle of containee. In Agreegate, container does not manage the lifecycle of the data.

Read the API docs for QList and find three distinct ways to add elements to the list.

void append (const T & value)
void prepend (const T & value)
void push_back (const T & value)
void push_front (const T & value)
QList<T> & operator<< (const QList<T> & other)
QList<T> & operator<< (const T & value)
QList<T> & operator+= (const QList<T> & other)
QList<T> & operator+= (const T & value)

List three methods that exist in QStringList but are not in QList.

QString join(const QString& separator) const
QStringList& replaceInStrings(const QRegExp& rx, const QString& after)
void sort()

Why does QList have an iterator and an Iterator? What is the difference between them?
Discuss the advantages and disadvantages of Qt or STL container classes as compared with arrays.
Discuss the syntax of the QList declaration. What should be in <angle brackets>? Why is it necessary?
Why does Qt support so many different kinds of iterators?
What is the difference between QFile and QFileInfo?

5.13. Review Questions

[fromfile: functions-questions.xml id: functions-questions]
What is the difference between a function declaration and a function definition?

A declaration is for describing how a function can be called. A definition includes the actual code that is to be executed.
We are declaring a type when we are writting it. We aredefining it instead
when we are putting the prototype and body.
Why are default argument specifiers in the declaration but not the definition?

Because it describes how the function can be called (interface), not how the function is implemented.
for every member function, it can have defaultl functions. Meaning you can provide standardvalues to one of the or multiple arguments. So if
user doesnt'set one of these arguments, then it willhave default value. Defalt values part ofthe definition.

Explain why is it an error to have two functions in the same scope with the same signature but with different return types.
For overloading arithmetic symbols (+ - * /) on Fraction objects, which is preferred, member functions or nonmember global operators? Explain your answer.

Nonmember global opreators are preferred because then the first or second operand can be converted just as easily to a Fraction. As a member function, only the right hand side is convertible, which means the operator won't work in a symmetric way.
For overloading left-modifying operators such as = and +=, which is preferred, member function operators or (nonmember) global operators?

Because you can and should return *this, which is not a reference to a temporary, a member function operator is the most appropriate. This permits chaining.
Explain the difference between pass-by-value and pass-by-reference. Why would you use one instead of the other?

A value parameter is a local variable in a function that holds a copy of a particular argument object that was passed to the function. Changes to that variable due to the action of the function have no effect on the argument object.

A reference parameter is an alias for a particular argument object that was passed to the function. No copy is made of the argument object. The alias is local to the function and, if it is not declared const, can permit the action of the function to change the argument object.

By default in C, they arepass by value. Every copy isinitialized on stack before calling the function. But in C++, wecanalso pass by reference.

Explain the difference between preprocessor macros and inline functions.

6.11. Review Questions

[fromfile: inheritance-questions.xml id: inheritance-questions]
What is the difference between a function and a method?

A method is a virtual function. A method call is resolved at runtime, a function call can be determined at compile-time.
due to if a function in class was a virtual, it wouldthus be amethod.
What does it mean for a base class function to be *hidden*? What can cause this to happen?

So, when a base class is derived from the parent class, you can override the parent's method. If you do override it, then the method of the parent's class is hidden, so you can't use it. If you call that method, you're going to call the base class's method.

Ex.
Parent class Bob with method eatFood().
```
Void Bob::eatFood(){
        1 + 1 = base;
}
```
Base class Marley is derived from Bob and eatFood() is overrided and does something else.
```
Void Marley::eatFood(){
        2+ 1 = base;
}
```

Calling marley.eatFood() would call marley's eatFood and not Bob's eatFood. So Bob's eatFood is hidden.

Which member functions *cannot* be inherited from the base class? Explain why.

The constructors, destructors, copy and assignment operators are all not inherited. They get generated for classes depending on certain rules, and the generated versions call base class versions.

7.5. Review Questions

[fromfile: libraries-questions.xml id: libraries-questions]

What is a platform? What platform do you use? What platform can be found in the labs at your school or work place?

What is code reuse? How is it done? What is good about it?

What is the role of a compiler?

What is the role of a linker?

Name three kinds of C++ libraries.

What is CPPLIBS? Why do you need it?

For each of these items, decide whether it would normally be found in a header (.h) file or an implementation (.cpp) file and explain why.

Function definitions

implementation - they should be only compiled once, and not included by other modules that use it.

Function declarations

usually the header file, to refer to functions defined in other separate implementation files.

static object declarations

header files, like most declarations.

static object definitions

implementation file, just like functions. We do not wish to have redundant storage allocation for statics.

Class definitions

These go in the header file.

Class declarations

header file - usually they are forward declarations.

inline function definitions

header files - unlike regular functions which can be linked together, inline functions need to be fully defined before they can be used.

inline function declarations

Header file - Rarely used for member functions except when the declaration is separated from the definition - but kept in the same header file.

Default argument specifiers

header file - default arguments to a function changes the way the function is called. This is the realm of the header file.

What is the difference between a compile time dependency and a link time dependency?

A compile time dependency exists if we need to #include the header file in order to reuse the symbol. A link time dependency exists if we can get away with just a declaration of the symbol, which requires the linker to be able to find the referenced symbol.

What is a framework? Are you using one?

What is a design pattern? What do most design patterns have in common?

Most design patterns describe how to separate code by responsibility. Each has a pattern name, a description of the problem to which the pattern applies, a description of how the pattern solves the problem, and a discussion of the consequences of employing the pattern.

What is an AntiPattern?

8.9. Review Questions

[fromfile: qobject-questions.xml id: qobject-questions]

What does it mean when QObject A is the parent of QObject B?

B is managed by A, and B will be destroyed when A is.

Means that specifiiues at run time who is pointing to

Which QObjects do not need a parent?

Those that are allocated on the stack.

What happens to a QObject when it is reparented?

It is removed from its former parent's child list, and added to the new parent's child list.

Why is the copy constructor of QObject not public?

To prevent it from being copied. QObjects are supposed to have identity, and so there should be no "exact" copy.

What is the composite pattern?
How can QObject be both composite and component?

When it has both parents *and* children.

How can you access the children of a QObject?
What is an event loop? How is it initiated?
What is a signal? How do you call one?

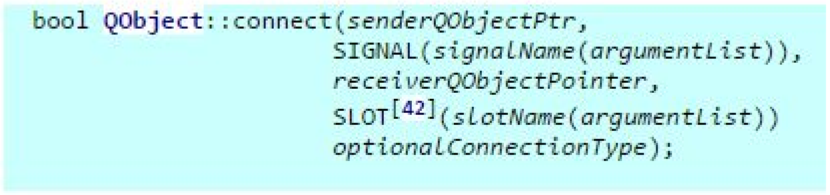
A **signal** is a message presented in a class definition like a void function declaration. It has a parameter list but no function body.

What is a slot? How do you call one?

- A **slot** is usually a void member function.
- It can be called as a normal member function

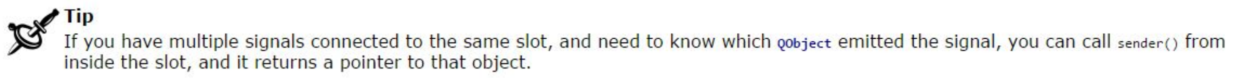
How are signals and slots connected?

- The syntax of the `connect` statement is



```
bool QObject::connect(senderQObjectPtr,
                      SIGNAL(signalName(argumentList)),
                      receiverQObjectPointer,
                      SLOT[42](slotName(argumentList))
                      optionalConnectionType);
```

In the case where multiple signals are connected to the same slot, how can you determine the QObject that emitted the signal?



Tip
If you have multiple signals connected to the same slot, and need to know which `QObject` emitted the signal, you can call `sender()` from inside the slot, and it returns a pointer to that object.

How can information be transmitted from one object to another?

- A typical Qt program creates objects, connects them, and then tells the application to `exec()`.
- At that point, the application enters the **event loop**.

Deriving a class from QObject more than once can cause problems. How might that happen accidentally?
What is the difference between value types and object types? Give examples.

- Instances of **value types** are usually relatively simple, occupy contiguous memory space, and can be copied or compared quickly.
- Examples of value types: *Anything**, `int`, `char`, `QString`, `QDate`, and `QVariant`.

- Any class that has a `public` default constructor, copy constructor, and copy assignment operator is a value type.
- Instances of **object types**, on the other hand, are typically more complex and maintain some sort of identity.

9.11. Review Questions

[fromfile: widgets-questions.xml id: widgets-questions]
List six things that QWidgets have in common.

They are QObjects, so they can have parents, children, properties, signals, and slots. They are QPaintDevices, so they have a height, width, depth, and can be painted on the screen.
What is a dialog? Where is an appropriate place to use it?
What is a QLayout? What is its purpose? What is an example of a concrete QLayout class?

A QLayout is a class whose sole purpose it is to arrange other QWidgets or sub-layouts. It has no visual representation itself. There are Boxes, Grids and Stacks.
Can a widget be a child of a layout?

When a widget is added to a layout, its parent is changed, but to the parent of the layout, not the layout itself. So the answer is "no".
Can a layout be a child of a widget?

Yes if it is a top-level layout.
Can a layout be a child of another layout?

Yes, that is how to nest layouts.
What are the advantages of listing your images in a resources file?

The images can get compiled into the executable and no longer need to be installed separately.
What is the difference between a spacer and a stretch?

A spacer is a fixed amount of space that does not stretch, while a stretch starts with a certain amount of space and expands to gobble up available space. In the case of multiple stretches in the same layout, the ratio can be used to determine which stretch expands faster.

10.8. Review Questions

[fromfile: mainwindows-questions.xml id: mainwindows-questions]
What are the main features of a QMainWindow?
QMainWindow and Dialog both inherit from QWidget. QMainWindow is divided in multiple
How can you install a QMenuBar in a QMainWindow?
How can you save and later restore the size, position, and arrangements of widgets for a GUI app?
What is a central widget?
How does a widget become the central widget?
What are dock widgets for?
How many dock widgets can you have in one application?
How can you make use of dock widgets?
What is an action?
How can you make use of actions?
What is an action group? Why would you have one in your application?
How can you make your application available to people who can't understand English?
What is a QAction? How are actions triggered?

A QAction is an object representing something that the user can do to the program. It can be triggered through menu items, button clicking, keyboard presses, etc.
It is possible to create QMenus without using QActions. What are the advantages of using a QAction?

By creating a unique QAction for each action, you can add it to different menus and toolbars, and from one place, disable/enable it, or change other properties (such as its label or tooltip) and it will reflect in all views (menuitems, toolbar buttons, etc) that refer to it.
Which Qt classes use the Command pattern?

QRunnable, QProcess, QAction, QThread, QtConcurrentRun

11.7. Review Questions

[fromfile: generics-questions.xml id: generics-questions]
Explain an important difference between a template parameter and a function parameter.

A function parameter must be a constant or a variable, while a template parameter can also be (and most commonly is) a *type expression* such as int, QObject, Person*.
What does it mean to instantiate a template function? Describe one way to do this.

Calling a template function with actual arguments is one way to instantiate a template function. In general, using a template definition, by providing actual template parameters, is how you can get the compiler to generate the template code for the specific type you are using.

when we define template. We define template parameters. Can be multiple and cann be type but also constant. instead the param of function can only be tyes.
Normally, you need to place template definitions in header files. Why is this?

Because, like any inline functions, because the compiler must substitute the definition in the code where the invocation is encountered, the definition must be known by the compiler. The linker will not resolve it later. The exception to this is with compilers which support export and maintain a database of template instantiations which get built along with the object modules. Such compilers blur the distinction between a compiler and a linker, but c'est la vie.
this is because to use same template, then you get an error.
Some compilers support export. What is it for?

Export makes it possible for template definitions in one source module to be used in others, linked in a linker-like fashion. It helps reduce the code-size of a program, and also eliminates redundant code.
Qt's container classes are used to collect value types. What kinds of things are not appropriate to store by value in a value collection?

QObjects, polymorphic collections in general, and any class that can not be copied because it has no copy constructor.
Which containers provide a mapping from key to value? List and describe at least two, and tell how they are different from one another.

QMap and QHash.
What does it mean for a container to "manage" its heap objects? How can a container of pointers to heap objects become a "managed container"?
Give at least three examples of Qt classes that implement the Flyweight pattern.

QList, QString, QSet.
When defining a class template, how should the code be distributed between the header (.h) file and the implementation (.cpp) file? Explain your answer.

Template definitions (classes and functions) must all be located in the header file. This is necessary for the compiler to generate code from a template declaration. Also the template declaration code template<class T> must precede each class or function definition that has a template parameter in its name.

12.9. Review Questions

[fromfile: reflection-questions.xml id: reflection-questions]
What kinds of information can you obtain from a QMetaObject?

className(), which returns the class name as a const char*

superClass(), which returns a pointer to the QMetaObject of the base class if there is one (or 0 if there is not)

methodCount(), which returns the number of member functions of the class

method(index), which returns the meta-data for the method with the given index.

propertyCount(), which returns the number of properties in this class, including base class properties.

property(index), which returns the meta-data for the property with the given index.

Very immportant is MainobjectFramework. Divded into two parts. One part allows you to know weverything about QObjectType and the other about QValue Type.Like this because in QT, our types are divded into two categories. One has value types so you can clone and the other being dynamic from QObject. Everything inherited from QObject is not clonable. Only one instance.

ObjectType and ValueType. Different types. ValueTypes can be copied.

What Qt classes are used to do data reflection? Explain why.

QMetaObject, QMetaProperty, QSqlDatabase::tables, QSqlRecord, QVariant
How does the QMetaObject code for each of your QObject-derived classes get generated?

moc generates QMetaObject classes to support properties, signals and slots. Normally, you do not run moc directly. It is run automatically by make on the header files listed in HEADERS which use the Q_OBJECT macro.

What is a downcast? In what situations do you use them?

A downcast is a conversion from something "higher" (more general) in the inheritance graph, to a "lower" class (one of its derived types). We use downcasts in situations where we are forced to use a base class pointer (usually because a function we did not write is using that type) but we require a non-polymorphic derived class member function.

What is RTTI? How does Qt provide RTTI?

Discuss the differences between the dynamic_cast and qobject_cast operators.

What is a QVariant? How might you make use of one?

What are the advantages of using property() and setProperty() over direct getters and setters?

Q_PROPERTY macros make it possible for moc to generate code for QObject's property() and setProperty() member functions. The advantage of using these functions is that client code can determine which properties are available by iterating through QMetaProperty objects of the QMetaObject corresponding to that class.

What does the property() function return? How do you obtain the actual stored value?

property() returns a QVariant, which is a union wrapper around every possible basic type, and also several Qt classes/types. With QVariant, you can ask for its type() and convert to the actual value<>(). Benefits are most apparent when implementing script engines or developer tools. It becomes possible to define "handle anything" kinds of functions without using anything but the QObject interface to read and write values.

Explain how it is possible to add new properties, acquired at runtime, to a QObject.

setProperty("propName") sets a dynamic property for that object even if it is not declared as a Q_PROPERTY

Explain how dynamic properties can be serialized.

They are stored in a QVariantMap, which can be serialized via a QDataStream.

13.7. Review Questions

[fromfile: modelview-questions.xml id: modelview-questions]

What are model classes? What are view classes? What relationship should be maintained between them?
What tools are provided by Qt to work with models and views?
What is MVC?
What is controller code? Which Qt classes are controller classes?

Code which creates, destroys, or connects objects together is considered controller code. QApplication and QAction are both containers for controller code, but the ItemDelegate is also considered a controller.
What are delegates? Where are they found?
In relation to delegates, what are roles?
How do you determine what item(s) is/are selected in a QListView?

Each view has a SelectionModel which can describe which item(s) are selected in a view.
If you want to iterate through items in an QAbstractItemModel, what would be a good class to use?

QModelIndex is a "cursor", or an indirect reference to data in a model. This can be used to iterate through items in the model.
There are two hierarchies of classes for storing and displaying tree-like data: *Widget/Item and *ItemModel/View. What reasons might you have for using one rather than the other?
Why would you use the QStandardItemModel rather than the QAbstractItemModel? Or vice versa?

StandardItemModel holds onto the data, while an abstract model can be a proxy for data elsewhere. StandardItemModel can lead to duplication of data. It is recommended to implement abstract methods from an AbstractItemModel for custom models.

1.18.1
What is a stream? What kinds of streams are there?
The term stream is an abstraction of a construct that allows you to send or receive an unknown number of bytes.
A way to write to a file or console

1.18
Getline vs. >> operator?
Getline is a method of an object and >> is an operator.

1.18.6
Pointer vs reference?
Pointer is an address of memory possibly having data.
Pointer points to a data type.

Reference is an alias for another variable. You can use a reference as a normal variable.

1.18.8
What is the address-of operator?
&

1.18.9
Deference operator?
*
1.18.10
Null pointer?
The value of the pointer is 0. You want to use null pointers to help debug.

1.18.11
What is a memory leak?
Refer to online material

1.18.12
What might cause a segmentation fault?
A pointer pointing to something that does not exist.

1.18.14
Function's signature?

1.18.16
Why is it an error to have two functions with the same signature but different return types in one scope?

1.18.17
Why does main sometimes have parameters?

Chapter 2

2.16.1
Class vs struct?
Structs' members are all public.

2.16.2
Class scope vs. block scope?

2.16.4
Static vs. non-static?

Static members are shared among all instances of the class.

2.16.3
Friend function?
This function are able to access protected/private data members of the class.

2.16.6
To declare a member function to be *const*?
The function does not change any data members of the class.

2.16.7

Chapter 4

Iterator pattern?
Ex: Traversing through a list.
Powerful because the iterator does not need to know the data structure the class implements.

UML diagrams.

Aggregation vs. composition

Chapter 5

5.13.1
Function declaration and function definition?

5.13.2
Default arguments?
For every member function can have the full arguments.
Which are standard arguments (data types).
Definition does not have the default arguments.

5.13.6
Pass by value vs. Pass by reference
Pass by value - default in C, every argument is copied before calling the function
Pass by reference - use this when your data types are large (Ex: 1 GB because 1 GB will be copied to the stack)

Chapter 6

6.11.1
Method vs function?
Method is member function.

6.11.2
What does it mean to have a base class function to be hidden?
Functions in derived classes which don't override functions in base classes but which have the same name will *hide* other functions of the same name in the base class.

6.11.3
Which member functions cannot be inherited from the base class?
Constructor
Copy constructor

Chapter 7

Abstract class - class that cannot be instantiate directly. Defined by add a pure virtual member function.

Also to do not provide any public constructors. Define the constructor as private or protected.

Design pattern - good programming practice that helps to program the application to be usable and helps other p[people to better understand it
Ex: MVC

AntiPattern - is not a good practice of programming
Ex:

Chapter 8

8.9.1
QObject

What does it mean when QObject A is the parent of B?
When you create B you are specifying A as the parent. Parent-children relationship is at **run-time.** It specifies who is pointing to who. B is in the list of children of A.

8.9.4

Copy constructor is not public because?
Because every QObject is unique.

8.9.5
Composite patterns?

Event loops?
.exec opens up the event loop.

Signal?
Mechanism allows you to communicate between different objects. Run-time method. Exception safe. Signals and slots are used for communication between objects. The signals and slots mechanism is a central feature of Qt and probably the part that differs most from the features provided by other frameworks.

Slot?
Receiving the event from the signal.
A slot is a function that is called in response to a particular signal.

Connect signal and slot by using the connect() function.

Chapter 9 and 10

Qwidget and QMainWindow

Qwidget - main graphical object

QMainWindw - any object
Usually divided into multiple parts.
Ex: Menu with a status bar, left dock and right docks

QDialog - doesn't have all the multiple parts like QMainWindow

11.7.1
Difference between template parameter and a function parameter.

Every template is parametrized by one or more template parameters, indicated in the *parameter-list* of the template declaration syntax:

template < *parameter-list* **>** *declaration*	(1)

Each parameter in *parameter-list* may be:

- a non-type template parameter;
- a type template parameter;
- a template template parameter.

Non-type template parameter

type name(optional)	(1)	

type name(optional) = *default*	(2)	
type **...** *name*(optional)	(3)	(since C++11)

1) A non-type template parameter with an optional name.
2) A non-type template parameter with an optional name and a default value.
3) A non-type template parameter pack with an optional name.
type is one of the following types (optionally cv-qualified, the qualifiers are ignored):

- std::nullptr_t (since C++11);
- integral type;
- lvalue reference type (to object or to function);
- pointer type (to object or to function);
- pointer to member type (to member object or to member function);
- enumeration type.

Instantiate a template function?

Why place template definitions in header files?

If you want to use the template in many programs.

Chapter 12

QMetaObject

Two parts:

One part: you must know everything about your QObject types

To inherit from QObject (therefore not clonable)

Second part: you must know everything about your q value types
To clone them

Qvariant?

the difference between composition and aggregation

1. A "owns" B = Composition : B has no meaning or purpose in the system without A
2. A "uses" B = Aggregation : B exists independently (conceptually) from A

Example 1:

A Company is an aggregation of People. A Company is a composition of Accounts. When a Company ceases to do business its Accounts cease to exist but its People continue to exist.

Example 2: (very simplified)

A Text Editor owns a Buffer (composition). A Text Editor uses a File (aggregation). When the Text Editor is closed, the Buffer is destroyed but the File itself is not destroyed.

The differences between the "base class-subclass" relationship and the "parent- children classes" relationship

When a class is derived from a base class it is called **inheritance**. `You inherit when you want to add functionality to an existing component, or extend the component.`

When a class is referenced by/contained in a parent class it is called **encapsulation**. `You encapsulate when your (usually parent) object 'uses' components.`

`Exmaple:`

```
public class A{

}

public class AB : A{
}
```

class `AB` is derived class from `A`.

Parent and *Child* is a definition of *abstract* relationship, that in programming can get different shapes like:

```
public class A{

}

public class ParentA{

List<A> children = ...

}
```