**MATERIALE PDS**

Cosa trovo nelle classi evento, calendario e calendar list (gestite da google) ?

Classe evento: oggetto con titolo, start-end time, attendees.

eventi ricorrenti o eventi semplici.

Classe calendario: una collezione di eventi. ogni calendario

ha associato metadata, come calendar description o calendar time zone.

Calendar List: una lista di tutti i calendari di quell'utente.

Classe gestione bottoni qt:

<https://doc.qt.io/qt-5/qpushbutton.html>

Accesso account gmail:

email: toDoListAllInOne@gmail.com

password: pdspolito

API

https://developers.google.com/calendar/api/concepts/events-calendars

https://developers.google.com/calendar/api/v3/reference/calendars/

https://developers.google.com/calendar/api/v3/reference

<https://developers.google.com/calendar/api/guides/auth>

Documentazione api caldav calendar:

<https://developers.google.com/calendar/caldav/v2/guide?hl=it>

QT RICHESTE http

Le classi QNetwork servono per fare richieste http.

Per usarle bisogna prima aggiungere nel file CMakeLists.txt

find\_package(Qt6 COMPONENTS Network REQUIRED)

insieme agli altri "find\_package", e poi

target\_link\_libraries(prova\_pds PRIVATE Qt${QT\_VERSION\_MAJOR}::Network)

sotto l'altro "target\_link\_libraries" (in basso nel file).

------

Un esempio di uso delle classi QNetwork per fare una richiesta get al

sito "https://accounts.google.com/o/oauth2/auth":

int main(int argc, char \*argv[])

{

QApplication a(argc, argv); //c'Ã¨ sempre nel main

//prima si crea il manager

QNetworkAccessManager man;

//si crea l'oggetto richiesta specificando l'url

QNetworkRequest req(QUrl("https://accounts.google.com/o/oauth2/auth"));

//si ottiene la risposta inviando la richiesta tramite il manager

QNetworkReply\* reply =man.get(req);

//poi si scrive questa cosa cosÃ¬ com'Ã¨. Forse l'unica cosa su cui

//mettere mano Ã¨ l'ultimo parametro del metodo connect: una lambda

//expression per fare quello che vogliamo con la risposta ottenuta

QObject::connect(reply,&QNetworkReply::finished,[&](){

QByteArray read = reply->readAll(); //read contiene il contenuto della risposta

//si usa il metodo toStdString() per convertire read a stringa

std::cout << "Got response:\n" << read.toStdString() << std::endl;

reply->close();

reply->deleteLater();

a.quit();

});

/\*

MainWindow w;

w.show();

\*/

return a.exec();

}

-----

Per settare gli header si usa il metodo

void QNetworkRequest::setRawHeader(const QByteArray &headerName,

const QByteArray &headerValue)  
es

//creo la richiesta

QNetworkRequest req(QUrl("https://accounts.google.com/o/oauth2/auth"));

//imposto l'header authentication

req.setRawHeader(QByteArray("Authentication"),QByteArray("ciao"));

----

Utili per conversione da/a JSON

Generale:

https://doc.qt.io/qt-5/json.html

Oggetti:

https://doc.qt.io/qt-5/qjsonvalue.html

Array:

<https://doc.qt.io/qt-5/qjsonarray.html>