

Configuration Page

Cedrique Tassi & Andreas Haller

C++ Practice an der Universität Heidelberg

26. Januar 2016

Literatur

Stroustrup, Bjarne: *A Tour of C++*
Upper Saddle River, NJ: Addison-Wesley, 2014

Unzählige Internetseiten

Kapitel

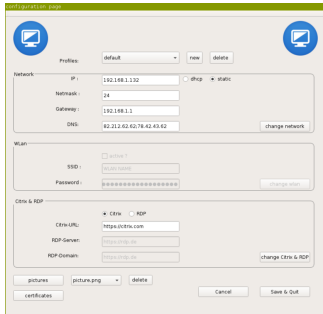
1 Problemstellung

2 Konzept & Realisierung

- Softwaremodulübersicht
- Implementierungsanforderungen
- Qt-Designer
- Setting & Profile
- Bilder & Zertifikate
- Fehlerbehandlung
- Netzwerk, WLAN, Citrix & RDP

3 Ausblick

4 Live-Demo



The screenshot shows a web-based configuration interface titled "Configuration Page" with the IT4S logo. It is divided into three main sections: Network, WLAN, and Citrix. The Network section has fields for Local IP (with a dropdown menu), Netmask (255.255.255.0), and Gateway (0.0.0.0), along with radio buttons for dhcp and static (selected). The WLAN section has an Active? checkbox, SSID (aewe), and Password (masked with asterisks). The Citrix section has a Store Link field (0.0.0.0). Each section has a "change" button, and there is a "quit" button at the bottom.

Configuration Page

IT4S

Network

Local IP: ☐ dhcp ☒ static

Netmask:

Gateway:

WLAN

Active? ☐

SSID:



Password:

Citrix

Store Link:

- Benutzerprofile & globale Einstellungen
- Netzwerktyp: dhcp/static
IP, Netmask, Gateway, DNS
- WLAN aktiv/inaktiv,
SSID, Passwort
- Verbindungstyp: Citrix/RDP
Citrix-Link, RDP-Server,
RDP-Domain
- Zertifikate & Firmenlogo
hinzufügen

configuration page



Profiles:

Network

IP: ☐ dhcp ☒ static

Netmask:

Gateway:

DNS:

Wlan

☐ active ?

SSID:

Password:

Citrix & RDP

☒ Citrix ☐ RDP

Citrix-URL:

RDP-Server:

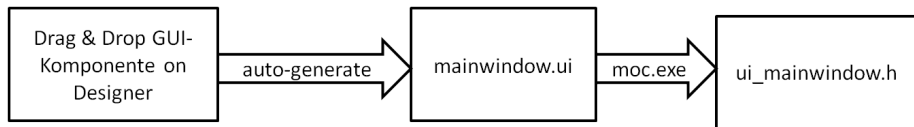
RDP-Domain:



Model View approach

- Aufbau von Qt-Klassen
- Erstellen von Dialogen
- SIGNAL und SLOT als Kommunikationsmechanismus
- Benutzen des Qt-Designers
- Robustheit und wiederverwendbarkeit

- Einfügen von GUI Komponenten per Drag&Drop
- Setzen von Komponenteneigenschaften
- Verbinden von Signalen
- Erstellen des ui-Files



Die Klasse MainWindow

```
class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();
    MainWindow(const MainWindow&) = delete;
    MainWindow* operator=(const MainWindow&) = delete;
    ...
private slots:
    void on_btn_cancel_clicked();
    void on_btn_save_quit_clicked();
    ...
private:
    Ui::MainWindow *ui;
    Setting setting;
    Profile profile;
};
```

- Abgeleitet von QMainWindow
- Kapselt alle GUI-Komponenten
(QWidget, QGroupBox, QComboBox etc. . .)
- Enthält zwei Member-Objekten, Member-Funktionen und Slots

Die Klasse Setting

```
class Setting
{
    private:
        QString mSettingsFile;
        QMap<QString, QMap<QString,QString> > mMap;
    public:
        Setting();
        void loadSettings();
        void saveSettings(QString profileName);
        QMap<QString, QMap<QString,QString> >& getSetting();
};
```

Der Member-Map dient zur Verwaltung der globalen Konfigurationsdatei

Globale Konfigurationsdatei (Setting.ini)

keys	Values
path_certificates	../ConfigPage/certificates/
path_client_logo	../ConfigPage/logo/client/
provider_logo	../ConfigPage/logo/provider/
path_usb	/media
path_log	../ConfigPage/log
path_profiles	../ConfigPage/profiles
last_profile	default
profile_opt	current_new
system	arch, debian
current_client_logo	

Die Klasse Profile

```
class Profile
{
    private:
        QString mProfilesFolder;
        QList<QString> mListOfProfilesName;
        QMap<QString, QMap<QString,QString> > mMap;
    public:
        Profile(void);
        Profile(QString profilesFolder);
        QString getProfileName();
        void setProfileFolderName(QString _profileFolderName);
        QString getProfileFolderName();
        void loadProfile(QString _profileName);
        QString saveProfile();
        QList<QString>& getListOfProfilesName(void);
        QMap<QString, QMap<QString,QString> >& getProfile();
};
```

Aufbau einer Profile-Datei

keys	Default-Values
citrix_rdp_citrix	https://citrix.com
citrix_rdp_rdp_domain	https://rdp.de
citrix_rdp_rdp_server	https://rdp.de
citrix_rdp_type	citrix
profile_name	default
network_dns	8.8.8.8
network_gateway	192.10.100.1
network_netmask	255.255.255.0
network_type	static
wlan_active	true
wlan_passwd	1234>?
wlan_ssid	"WLAN; NA"

Auszug aus der Klasse FileSystemModelDialog

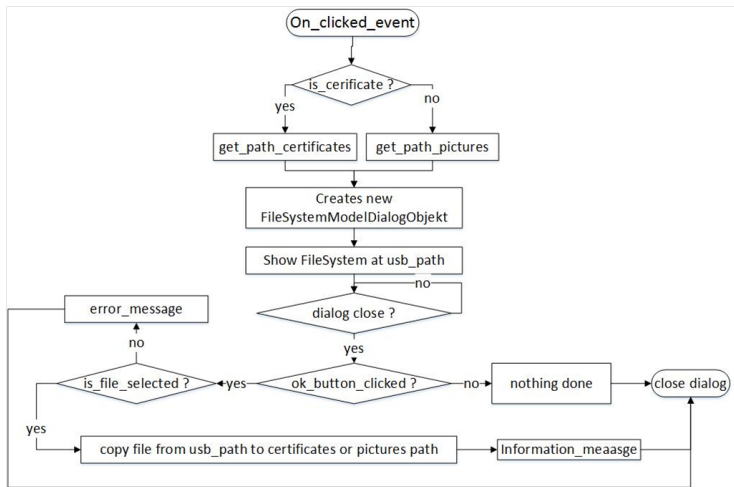
```
class FileSystemModelDialog : public QDialog
{
    Q_OBJECT

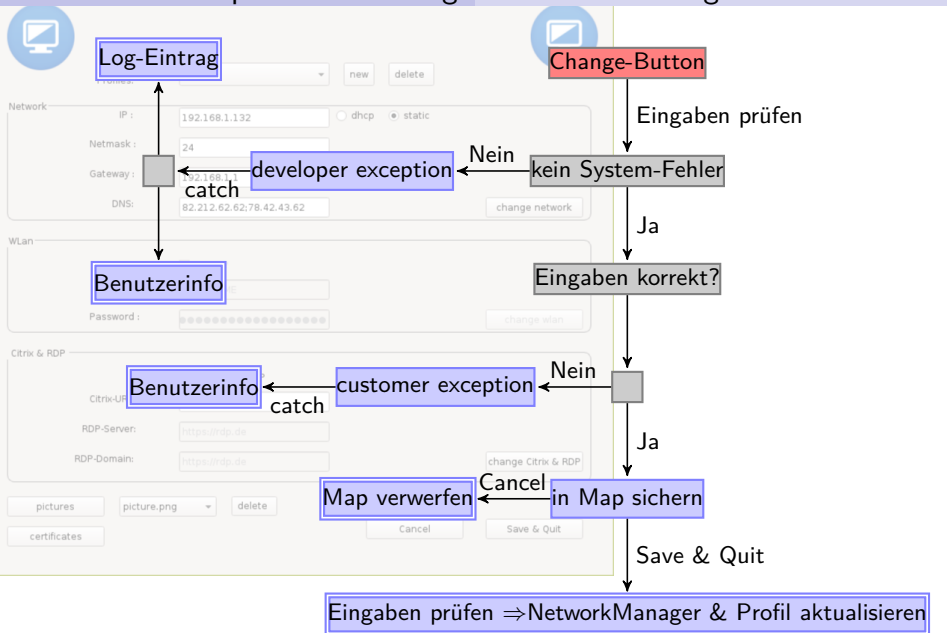
public:
    FileSystemModelDialog(const FileSystemModelDialog&) = delete;
    FileSystemModelDialog* operator=(const FileSystemModelDialog&) = delete;
    FileSystemModelDialog(QWidget *parent = 0,
                           const QString& usb_path = QString::null);
    ~FileSystemModelDialog();
private:
    QString mPath;
    QFileSystemModel *dirFilModel;
    QTreeView *treeView;
private slots:
    void accept();
    void reject();
};
```

Warum FileSystemModelDialog anstatt QFileDialog ?



- QFileDialog ermöglicht stets das Navigieren durch komplettes Dateisystem, selbst beim Festlegen des RootPathDir
- Nur usb_path sollte sichtbar sein
- ModelView mit QTreeView & QDirModel

Flussdiagramm zum Hochladen von Bildern & Zertifikaten





configuration page



Profiles:

Network

IP: ☐ dhcp ☒ static

Netmask:

Gateway:

DNS:

Wlan

☐ active ?

SSID:

Password:

Citrix & RDP

☒ Citrix ☐ RDP

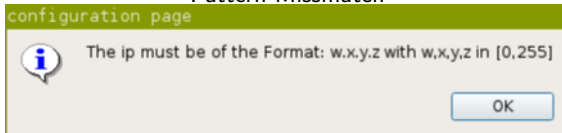
Citrix-URL:

RDP-Server:



RDP-Domain:

```
1 //ip-address must look like between 0.0.0.0 or 123.456.789.0
2 std::regex pat {R"^(^(\d{1,3}\.){3}\d{1,3}$)"};
3 if (std::regex_match(temp, pat)) {
4     //make more checks
5 } else {
6     throw customer_error(
7         std::string("The address must be of the Format: w.x.y.z ←
8             with w,x,y,z in [0,255]")
9     );
10 }
```

Pattern-Mismatch



configuration page



Profiles:

Network

IP: ☐ dhcp ☒ static

Netmask:

Gateway:

DNS:

Wlan

☐ active ?

SSID:

Password:

Citrix & RDP

☒ Citrix ☐ RDP

Citrix-URL:

RDP-Server:

RDP-Domain:

Ausführen von Konsolenbefehlen:

```
1 // unique_ptr with custom deleter function for FILE*: pclose().
2 //EXPLANATION
3 unique_ptr<FILE,           //raw pointer type: FILE*
4 int (*)(FILE*)>           //type: pclose() prototype
5 myFile(popen("myfile", "r"), // (FILE*) is returned by popen()
6 pclose );                 //deleter function: pclose()
7
8 //USEAGE:
9 unique_ptr <FILE, int (*) (FILE *)> file(popen(cmd, "r"), pclose);
10 output = read_out_stream(move(file));
11
12 //function prototype using this pointer
13 string read_out_stream(unique_ptr <FILE, int (*) (FILE *)> fp)
```

- Einstellung der systemweiten Sprache und des Tastaturlayouts
- Menü zur Sprache-Konfiguration
- Datenbank zur Profile-Speicherung

Dankeschön

Weitere Informationen sind im Wiki enthalten
{R"(Bitte [(?stellen Sie)|(?stell)]{1} offene Fragen nach der
Live-Demo)?"}