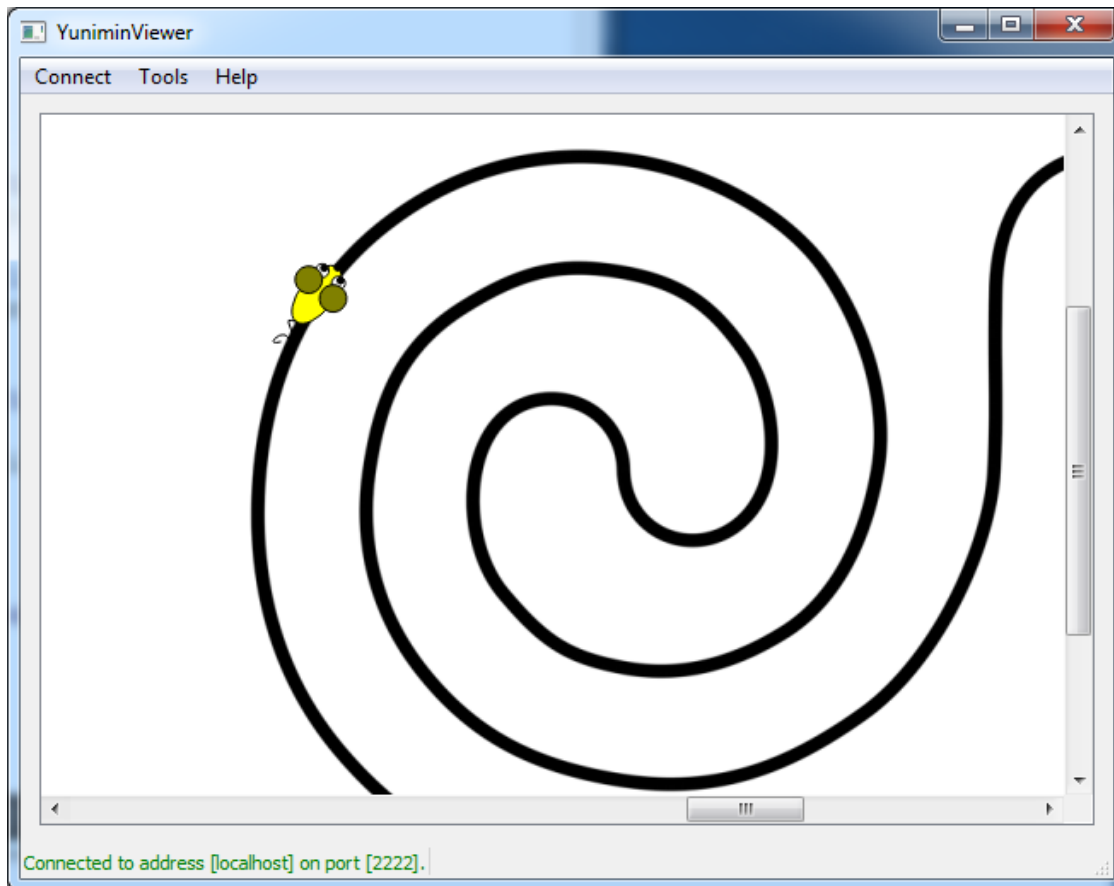


# 1 A teď něco prakticky

V této kapitole si vyzkoušíme rozpohybovat robota. Nejprve si doma zkusíte naprogramovat robota ve virtuálním simulátoru a pak si v rámci T-exkurze pohrajeme s reálným robotem.



Obrázek 1: Ukázka simulátoru v akci

Jako virtuální simulátor využijeme aplikaci, kterou naprogramoval Bedřich Said v rámci svojí práce SOČ. Její název je [Simulátor Yunimin](#) (pozn. název vznikl podle našeho nejstaršího výukového robota Yunimin, simulátor je ale univerzální a proto mi budeme využívat rozhraní pro robot Pololu 3pi, kterého máme na Robotárně minimálně v 10 kusech).

## 1.1 Jak rozběhnout simulátor

Pro zprovoznění simulátoru budete nejprve potřebovat nainstalovat vývojové prostředí pro Qt framework. Qt framework je soubor knihoven pro multiplatformní vývoj. Umožňuje vám naprogramovat aplikaci, která prakticky bez jakýchkoliv úprav bude fungovat jak na Windows, tak na Linuxu nebo Macu. To je taktéž jeden z důvodů, proč využíváme toto vývojové prostředí. Bohužel momentálně máme problém s jednou částí simulátoru a proto jej lze zatím provozovat jen na Windows.

### 1.1.1 Instalace prostředí Qt

Stáhněte si vývojové prostředí pro Qt (Qt Creator) ze [stránek výrobce](#). Zvolte si vhodnou verzi dle vašeho operačního systému (32/64-bit) a nainstalujte jej. Nejlepší volba pro vás pravděpodobně bude [Qt 5.6.0 for Windows 64-bit \(VS 2015, 836 MB\)](#).

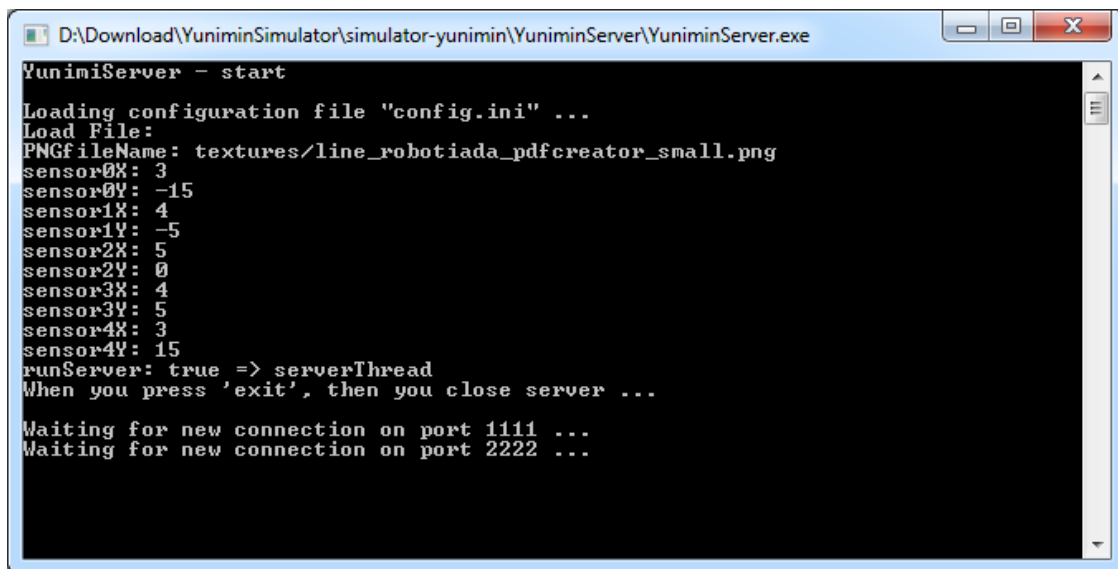
### 1.1.2 Stažení simulátoru

Po stažení a nainstalování prostředí si musíte [stáhnout Simulátor Yunimin](#). Vyberte nejaktuálnější archiv (při psaní návodu to byla [SimulatorYunimin1.0.3.zip](#)). Po stažení archivu jej rozbalte do vašeho pracovního adresáře.

### 1.1.3 Spuštění simulátoru

Simulátor se skládá ze tří samostatných aplikací. Základní stavebním kamenem je server, na kterém běží samotná simulace. Vy jako uživatel si můžete průběh simulace prohlížet pomocí vieweru. Server i viewer spustíte jako samostatné .EXE soubory.

#### Server



```
YunimiServer - start
Loading configuration file "config.ini" ...
Load File:
PNGfileName: textures/line_robotiada_pdfcreator_small.png
sensor0X: 3
sensor0Y: -15
sensor1X: 4
sensor1Y: -5
sensor2X: 5
sensor2Y: 0
sensor3X: 4
sensor3Y: 5
sensor4X: 3
sensor4Y: 15
runServer: true => serverThread
When you press 'exit', then you close server ...
Waiting for new connection on port 1111 ...
Waiting for new connection on port 2222 ...
```

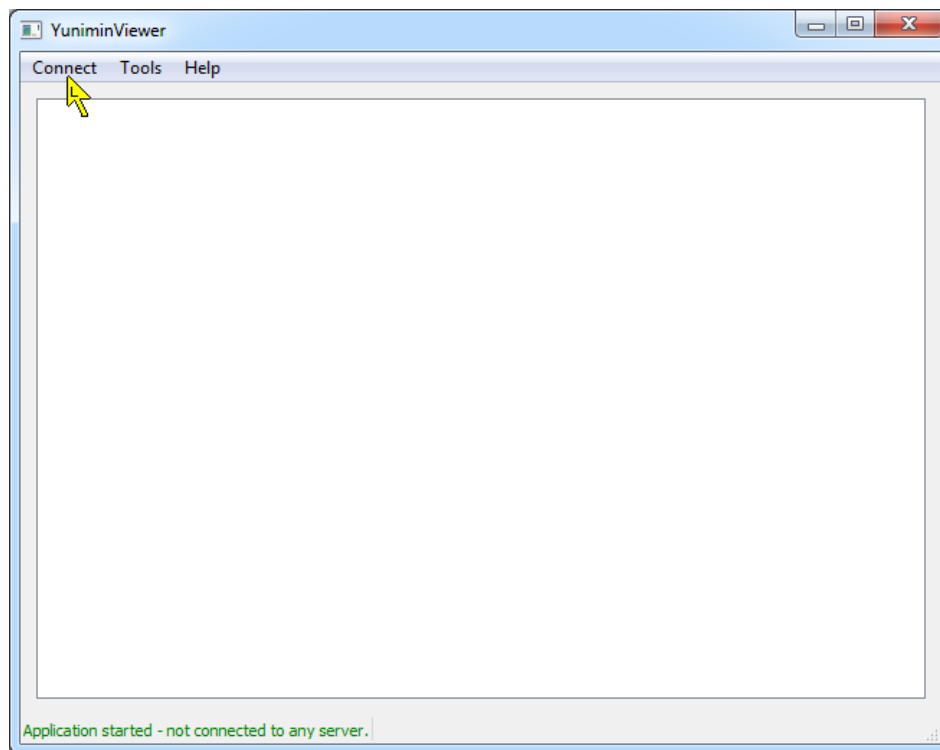
Obrázek 2: Terminál se spuštěným serverem

Pro spuštění serveru otevřete složku `YuniminServer` ve staženém archivu a spusťte soubor `YuniminServer.exe`. Po spuštění by se měl objevit terminál (Command-line interface), na kterém uvidíte informace ze serveru.

#### Viewer

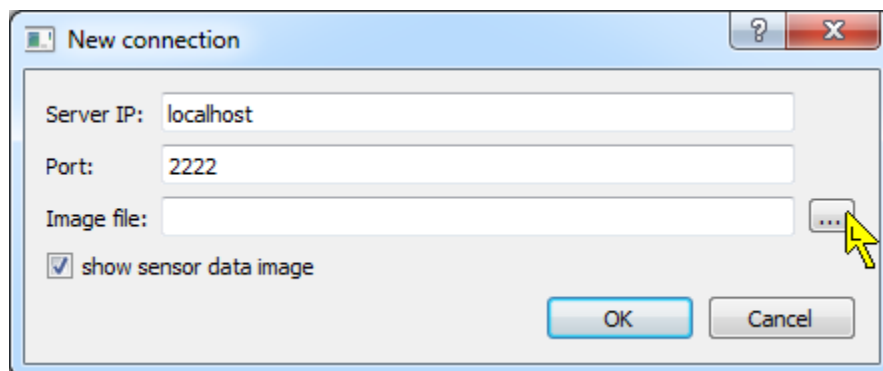
Viewer spustíte podobně jako server. Je potřeba otevřít složku `YuniminViewer` a v ní spustit soubor `YuniminViewer.exe`. Otevře se vám okno zobrazující simulátor.

Po startu programu je ovšem ještě třeba viewer připojit k serveru (tyto dvě aplikace běží zcela nezávisle a například server může běžet úplně někde jinde než viewer nebo klient). To provedeme tak, že si v



Obrázek 3: YuniminViewer po startu

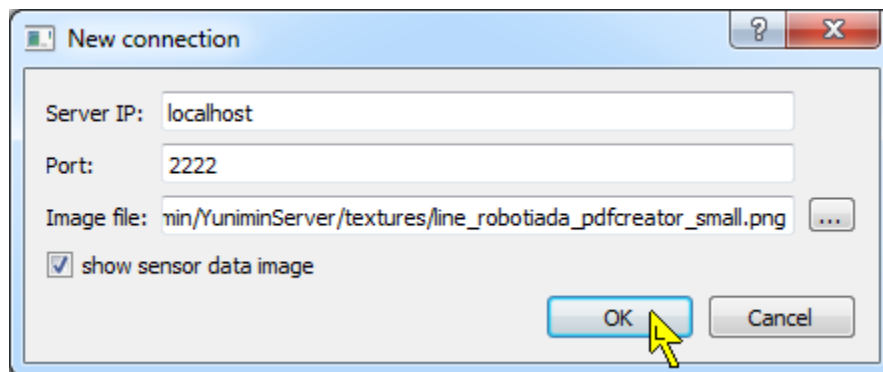
horní nabídce otevřeme **Connect** a vybereme **Connect to server**. Následně se nám zobrazí okno s nastavením připojení.



Obrázek 4: YuniminViewer - okno s nastavením připojení

Jelikož nám server běží lokálně na našem počítači (neběží jako veřejný server, ke kterému by se dalo přistupovat přes internet) ponecháme **Server IP** a **Port** tak jak jsou přednastaveny. Pro správné zobrazení ovšem potřebujeme načíst **Image file**, který je v serveru nastaven jako podklad a podle kterého nám robot bude vracet hodnoty podkladu pod jeho senzory.

Rozklikneme tedy nabídku pro výběr cesty k souboru (...) a otevřeme soubor, který je umístěn v `YuniminServer\textures\line_robotiada_pdfcreator_small.png` (jedná se o tréninkové hřiště pro jízdu s robotem po čáře) a potvrdíme připojení.



Obrázek 5: YuniminViewer - cesta k souboru s podkladem

Nyní je již viewer připojen k serveru a zobrazuje aktuální stav simulátoru. Ovšem v simulátoru se teď nic neděje, protože jsme ještě nepřipojili našeho robota.

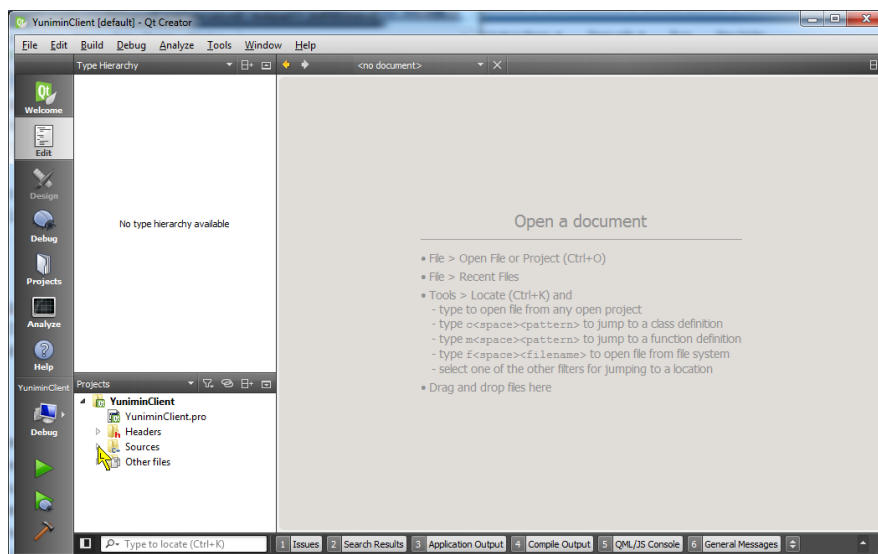


Obrázek 6: YuniminViewer po načtení souboru a připojení k serveru

## Client

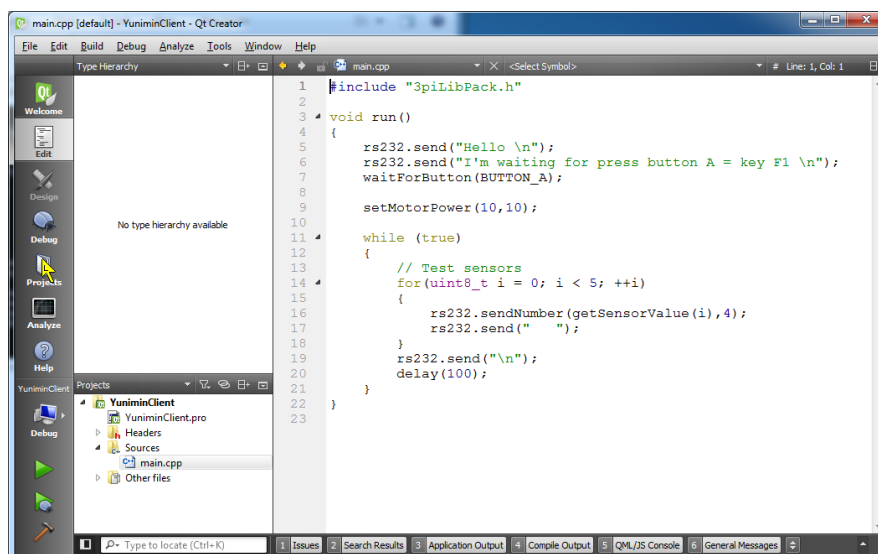
Váš robot bude simulován pomocí klienta. Jeden klient připojený k serveru odpovídá jednomu simulovanému robotovi. Program, který napíšete, bude zkompilován (převeden do spustitelného programu) jako součást klienta. To je důvod, proč jsme si instalovali prostředí Qt, které nám jednoduše program zkompiluje. Qt Creator spustíte tak, že ve složce **YuniminClient** si otevřete soubor **YuniminClient.pro**. Po otevření tohoto souboru by vám měl naběhnout Qt Creator s otevřeným projektem **YuniminClient**.

V projektu rozklikneme složku **Source** a poklepáním otevřeme soubor **main.cpp**.



Obrázek 7: Otevřený QT Creator s projektem YuniminClient

Nyní byste měli vidět zdrojový kód vašeho robota s předpřipraveným ukázkovým kódem předvádějící práci s komunikační linkou, tlačítka, motory a senzory.

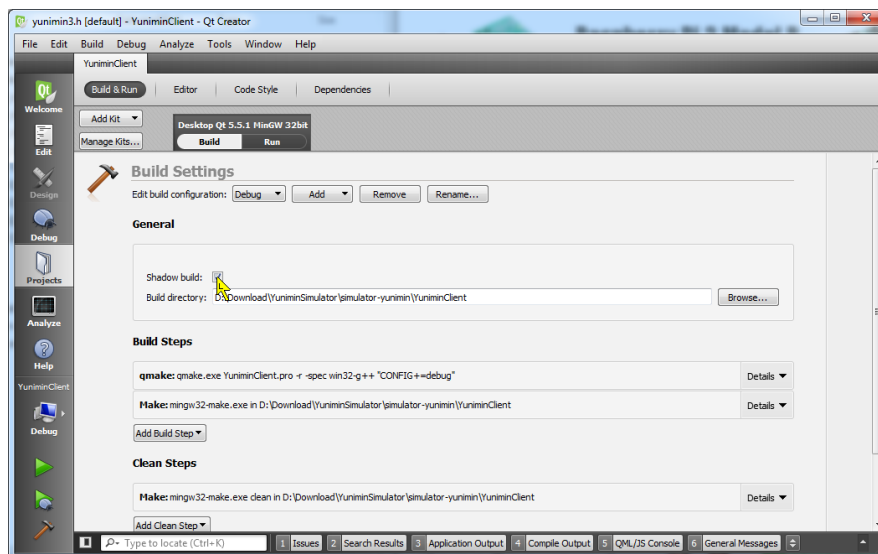


Obrázek 8: Otevřený projektem YuniminClient - main.cpp

Za chvíli si tento program zkusíme spustit, ale ještě před tím musíme zkontrolovat nastavení projektu. Proto si otevřeme nabídku **Projects** v levé boční liště (viz žlutá šipka v obrázku 8).

V nabídce **Projects** lze nastavit prakticky vše k danému projektu. Od konfigurace kompilace až po nastavení editoru textu v Qt (jak chcete odsazovat text, jakou barvu mají mít jednotlivé konstrukce jazyka, kolik mezer má představovat jeden tabulátor atd.).

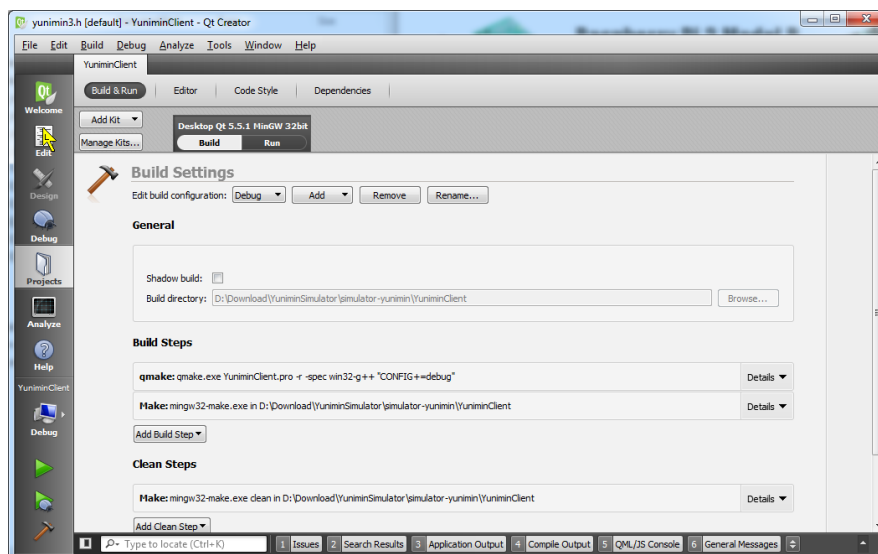
Po otevření nabídky musíte ověřit zda je deaktivované políčko **Shadow build**, které umožňuje kompi-



Obrázek 9: QT Creatoru - Project: deaktivace shadow build

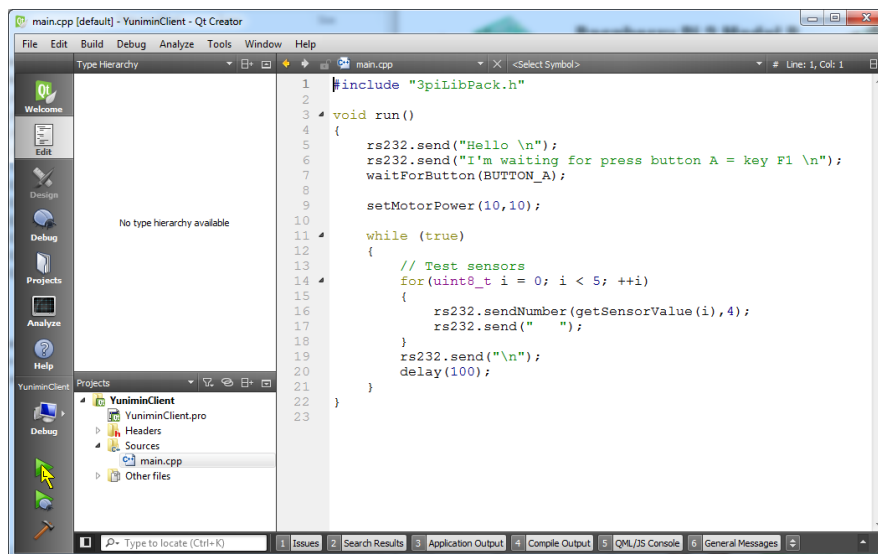
laci/buildování programu mimo adresář se zdrojovým kódem. My ale pro správnou funkčnost klienta potřebujeme zajistit kompilaci/buildování v rámci adresáře se zdrojovým kódem. Pokud je tedy políčko **Shadow build** aktivováno, tak jej kliknutím deaktivujte.

Nyní se můžeme vrátit do editoru pomocí ikonky **Edit** v levé boční liště a přejdeme ke spuštění našeho programu/robotu.



Obrázek 10: QT Creatoru - Project: návrat do editoru

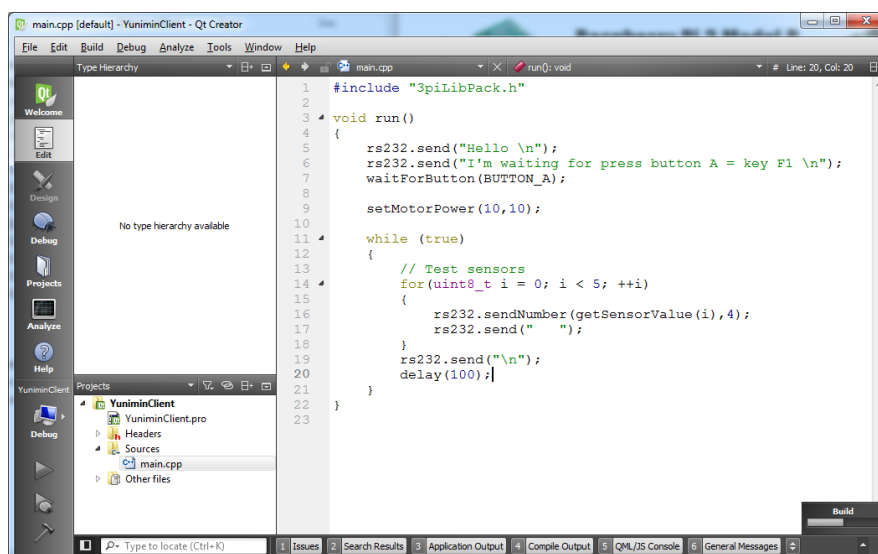
Pro spuštění programu je potřeba kliknout na zelenou šipku v levé boční liště (ta bez brouka :-)), případně můžete použít i klávesovou zkratku **CTRL + R**. Šipka s broukem slouží pro takzvaný debug režim. Díky tomuto režimu lze krokovat program po jednotlivých řádcích a zjišťovat, co se kde děje. Krokování využijete převážně při hledání chyb, což momentálně není náš případ.



Obrázek 11: QT Creatoru - spouštění programu

Po zmáčknutí zelené šipky začne probíhat Build, což nám indikuje ukazatel v pravém dolním rohu. Pokud proběhne vše v pořádku, tak se ukazatel zaplní zelenou barvou a program se spustí.

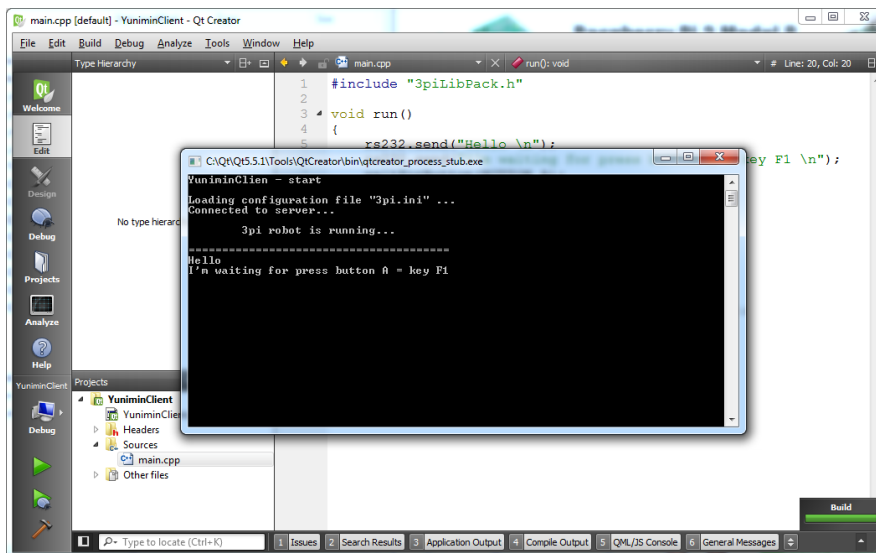
Pokud se program nespustil, tak QT Creator narazil při kompilaci/buildování na chybu, která neumožňuje vytvořit a spustit program. Ve spodní liště by se pak měla otevřít nabídka **Issues** a v ní by měli být vypsány všechny problémy nalezené při kompilaci. V případě, že na podobný problém narazíte, snažte se řešit problémy od prvního k poslednímu a pokaždé, když si myslíte, že jste odstranili alespoň jednu chybu, tak program znovu zkompilujte. Často se totiž stává, že jedna chyba generuje několik **Issues/Errorů**, tudíž po odstranění první chyby mohou zmizet i všechny další.



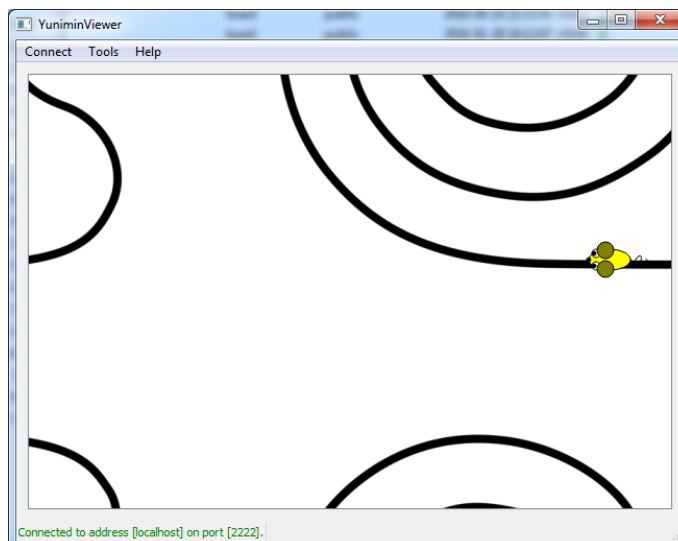
Obrázek 12: QT Creatoru - kompilace/buildování programu

Když program naběhne zobrazí se nám terminál. Pokud vše proběhne v pořádku, měli byste v něm

vidět přesně to stejné jako na obrázku 13. Znamená to, že program se spustil, připojil k serveru a již čeká jen na zmáčknutí tlačítka F1. Pokud se tak nestane, pravděpodobně není spuštěn server nebo se nepodařilo načíst konfigurační soubor (možná jste vynechali krok s Shadow build).



Obrázek 13: QT Creatoru - spuštěný program



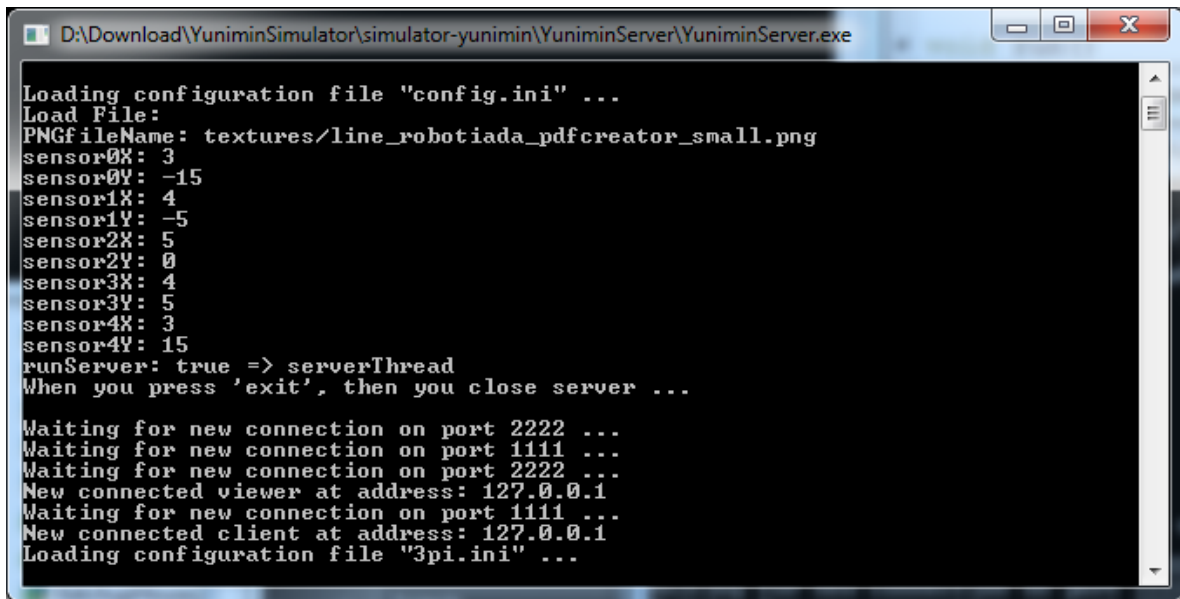
Obrázek 14: Viewer - spuštěný program

Ve vieweru se nyní objevila myš, která představuje vašeho robota. Vzhled robota je pevně nastaven a nedá se měnit.

Server by měl zobrazit připojení nového klienta (poslední dva řádky na obrázku 15).

Přejděte do terminálového okna klienta a zmáčkněte klávesu F1 (na reálném robotovi jsou umístěny tlačítka A, B, C - v rámci simulátoru jsou tyto tlačítka namapovány na klávesy F1 až F3). V terminálu se začnou vypisovat aktuální hodnoty ze senzorů (obr. 16). Ve vieweru lze sledovat pohyb tohoto robota (obr. 17). Robot by měl jet pořád doleva podél černé čáry.

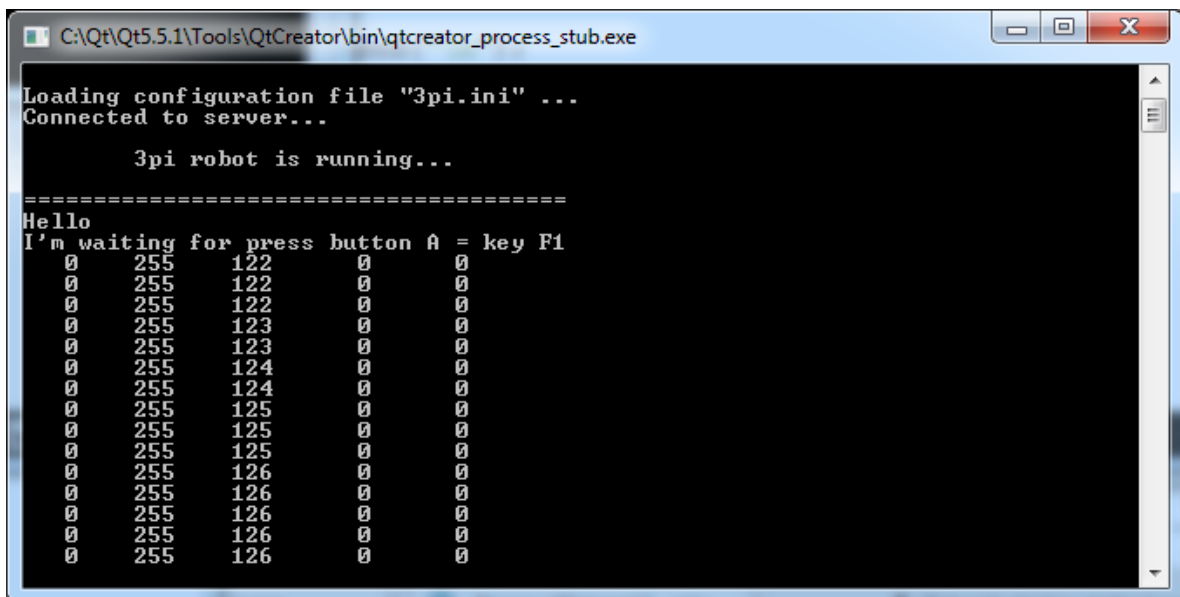




```
D:\Download\YuniminSimulator\simulator-yunimin\YuniminServer\YuniminServer.exe
Loading configuration file "config.ini" ...
Load File:
PNGfileName: textures/line_robotiada_pdfcreator_small.png
sensor0X: 3
sensor0Y: -15
sensor1X: 4
sensor1Y: -5
sensor2X: 5
sensor2Y: 0
sensor3X: 4
sensor3Y: 5
sensor4X: 3
sensor4Y: 15
runServer: true => serverThread
When you press 'exit', then you close server ...

Waiting for new connection on port 2222 ...
Waiting for new connection on port 1111 ...
Waiting for new connection on port 2222 ...
New connected viewer at address: 127.0.0.1
Waiting for new connection on port 1111 ...
New connected client at address: 127.0.0.1
Loading configuration file "3pi.ini" ...
```

Obrázek 15: Server - připojení nového klienta

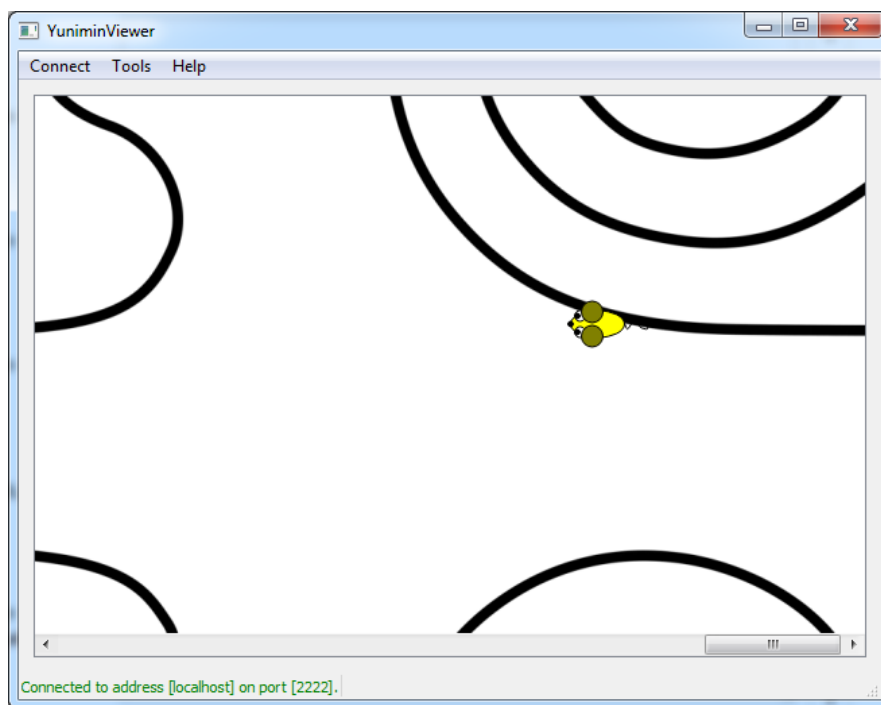


```
C:\Qt\Qt5.5.1\Tools\QtCreator\bin\qtcreator_process_stub.exe
Loading configuration file "3pi.ini" ...
Connected to server...

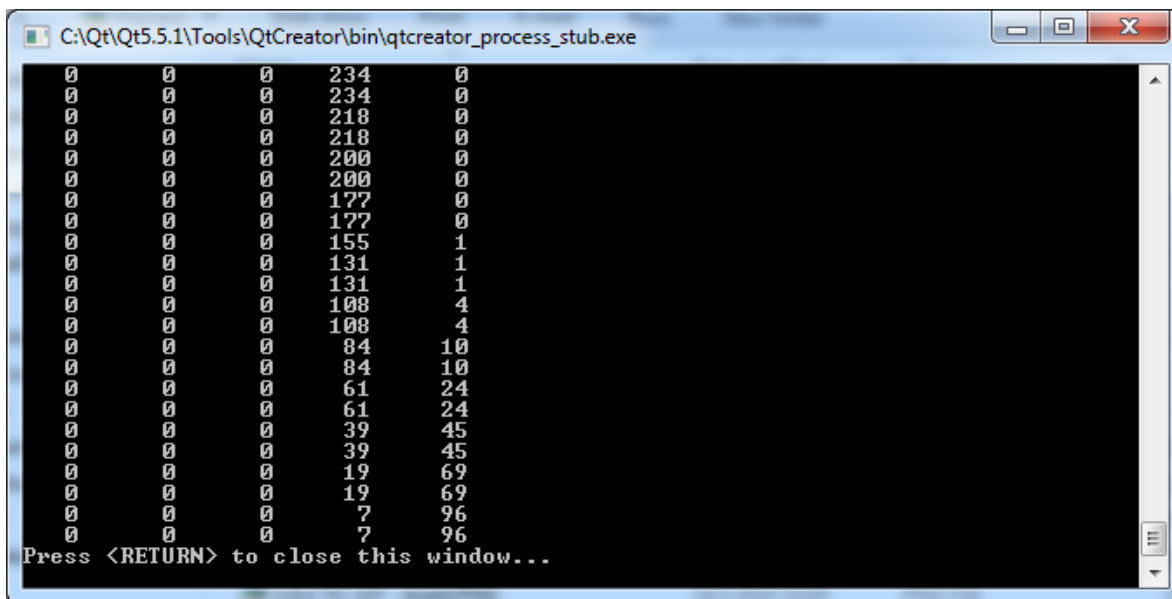
    3pi robot is running...

=====
Hello
I'm waiting for press button A = key F1
 0   255   122   0   0
 0   255   122   0   0
 0   255   122   0   0
 0   255   123   0   0
 0   255   123   0   0
 0   255   124   0   0
 0   255   124   0   0
 0   255   125   0   0
 0   255   125   0   0
 0   255   125   0   0
 0   255   126   0   0
 0   255   126   0   0
 0   255   126   0   0
 0   255   126   0   0
```

Obrázek 16: Client - program běží

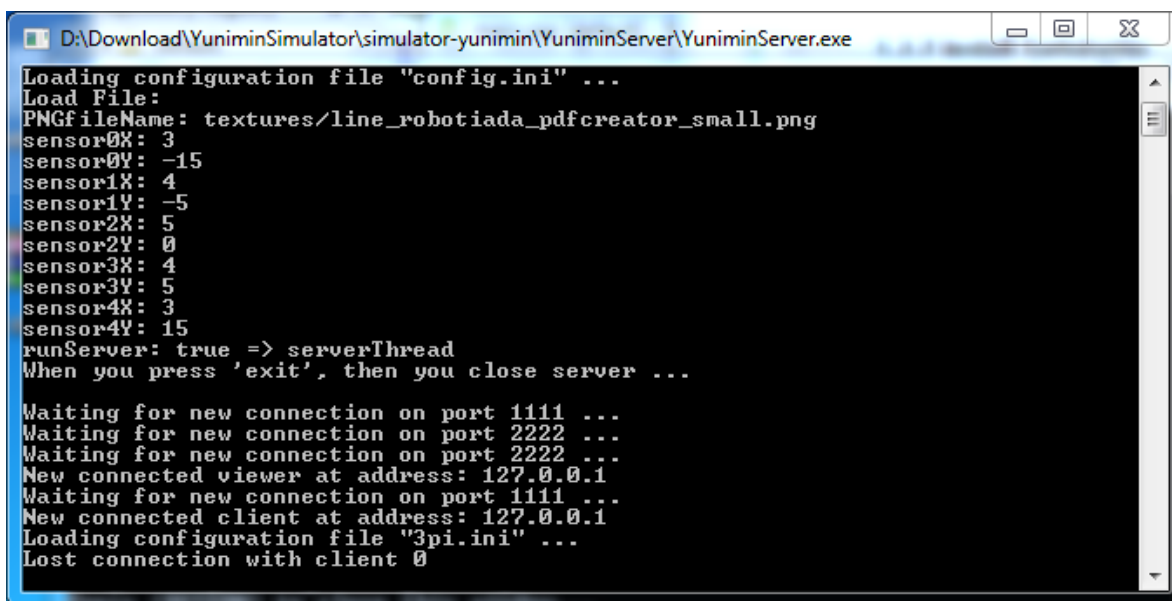


Obrázek 17: Viewer - program běží



Obrázek 18: Client - program zastaven

Program robota zastavíte tak, že se přepnete do klientova okna terminálu a zmáčknete klávesovou zkratku CTRL + C. Tím se ukončí vykonávání programu (obr. 18), klient se odpojí od serveru (obr. 19) a robot zmizí z viewru.



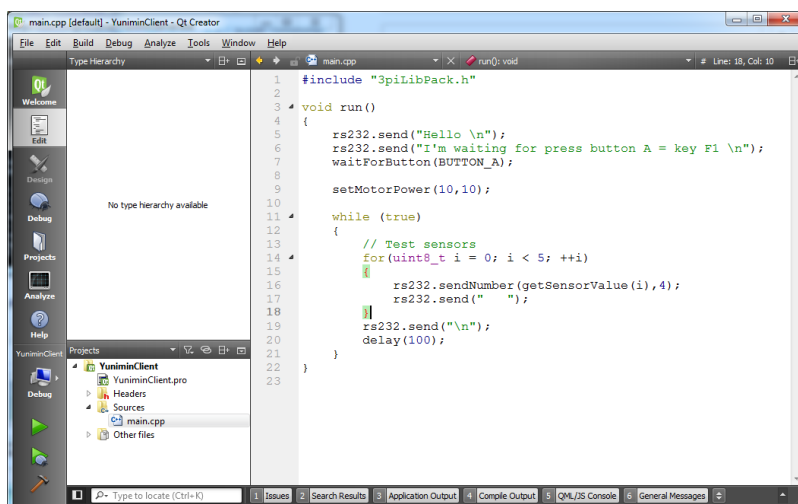
```

D:\Download\YuniminSimulator\simulator-yunimin\YuniminServer\YuniminServer.exe
Loading configuration file "config.ini" ...
Load File:
PNGfileName: textures/line_robotiada_pdfcreator_small.png
sensor0X: 3
sensor0Y: -15
sensor1X: 4
sensor1Y: -5
sensor2X: 5
sensor2Y: 0
sensor3X: 4
sensor3Y: 5
sensor4X: 3
sensor4Y: 15
runServer: true => serverThread
When you press 'exit', then you close server ...

Waiting for new connection on port 1111 ...
Waiting for new connection on port 2222 ...
Waiting for new connection on port 2222 ...
New connected viewer at address: 127.0.0.1
Waiting for new connection on port 1111 ...
New connected client at address: 127.0.0.1
Loading configuration file "3pi.ini" ...
Lost connection with client 0

```

Obrázek 19: Server - klient odpojen



Obrázek 20: QT Creator - můžete začít programovat

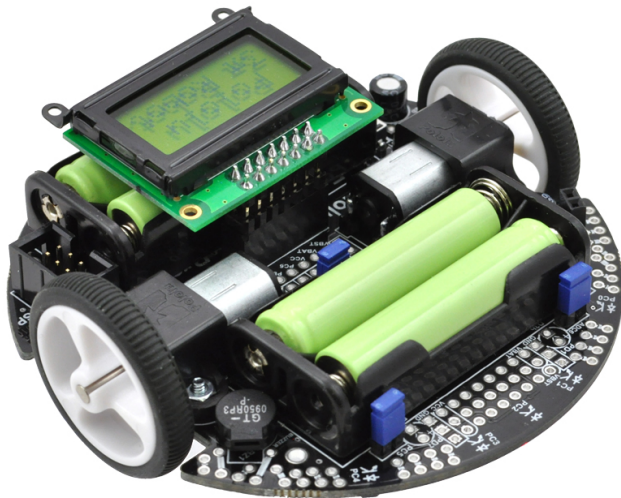
Nyní již víte jak obsluhovat simulátor a můžete tedy začít programovat robota.

## 1.2 Jak programovat robota

V rámci T-exkurze budeme pracovat s robotem **Pololu 3pi**, proto je simulátor nastaven tak, aby se chování robota, co nejvíce podobalo chování reálnému robotovi.

### 1.2.1 Popis robota Pololu 3pi

Robot Pololu 3pi je robot určen na rychlou jízdu po čáře. Zároveň je ale udělán tak, aby s ním mohl začít i nováček v oblasti programování mikrokontrolérů.



Obrázek 21: Robot Pololu 3pi

Jak již název napovídá, robot bude mít co dočinění s 3\*pi. Jeho průměr je totiž roven 3\*pi (cca. 9,5 cm). Robot je vybaven mikrokontrolérem Atmel ATmega328P, 5 senzory odrazivosti pro snímání podkladu, 3 tlačítka, 2 motory, displejem a bzučákem.

V simulátoru lze využít většinu součástí robota až na displej a bzučák, které zatím nejsou v simulátoru implementovány.

### 1.2.2 Programování robota

Pro robota Pololu 3pi nachystal Vojta Boček knihovnu, která usnadňuje jeho programování. Knihovna obsahuje prakticky vše, co robot může dělat. Dokumentaci knihovny naleznete na těchto webových stránkách: <https://github.com/Tassadar/3piLib/wiki>

Jak již ale bylo řečeno, simulátor nepodporuje displej a bzučák, takže tyto komponenty nelze v simulátoru využít.

### 1.2.3 Jak začít

Na začátek bude nejlepší upravit si zdrojový soubor `main.cpp` v klientovi tak, že odstraníte vše uvnitř funkce `void run()` a budete si postupně zkoušet jednotlivé funkce.

Zkuste si rozjet robota tak, aby jezdil do kruhu, pak můžete zkusit s robotem jezdit ve spirále, oběd obdélník, trojúhelník, měnit velikost jednotlivých stran (zkuste použít i složitější konstrukce jazyka C++ jako podmínka a cyklus).

Pro inspiraci přidáváme odkaz na ukázkou jednoduchých programů (možná bude třeba pro simulátor doladit časové konstanty - `delay()` v ukázkových programech): <http://robotikabrno.cz/robotika-brno/navody/robot-pololu-3pi>

### 1.3 Zadání úkolu

Až si projdete jednotlivé funkce a vyzkoušíte si, co vše s robotem lze dělat můžete přejít na řešení úkolu (není to tedy úplně podmínkou, lze úkol vyřešit i bez vyzkoušení si simulátoru, ale chtěli bychom aby jste si vyzkoušeli s robotem pracovat již doma a pak abychom již v rámci samotné T-exkurze mohli přejít ke složitějším věcem).

Vášim úkolem bude navrhnout program nebo algoritmus, díky kterému bude jezdit robot po čáře. Chceme jen základní algoritmus