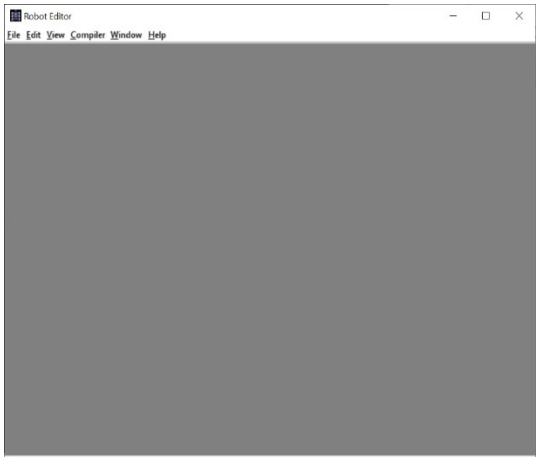
Erste Schritte mit Robocode

1 Bevor du beginnst

- Stelle sicher, dass Du Java und Robocode wie im Dokument vom ersten Tag **Robocode auf Windows installieren.docx** beschrieben, installiert hast.

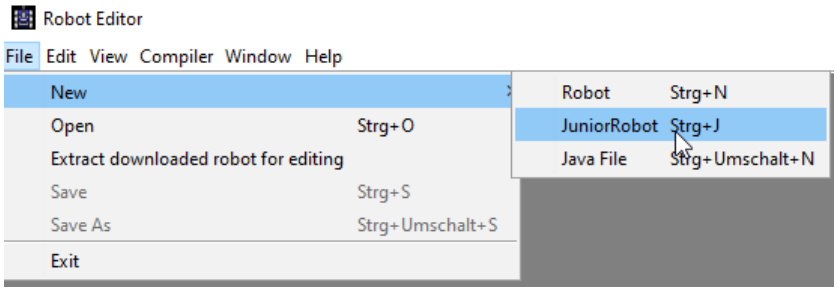
2 Starte Robocode und verwende den Code-Editor

- Starte jetzt Robocode wie es im Dokument vom ersten Tag, **Robocode auf Windows installieren.docx**, beschrieben ist.
- Jetzt wähle vom **`Robot Menu`** den Source Editor (Quellencode Editor) aus, um den Java Editor zu starten. Auch das ist im Dokument von gestern beschrieben.

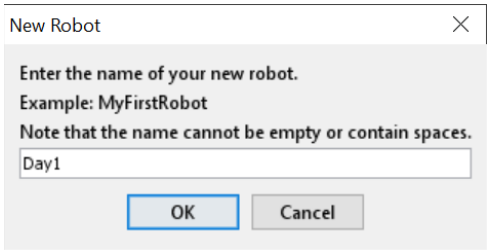


3 Erstelle Deinen ersten Roboter

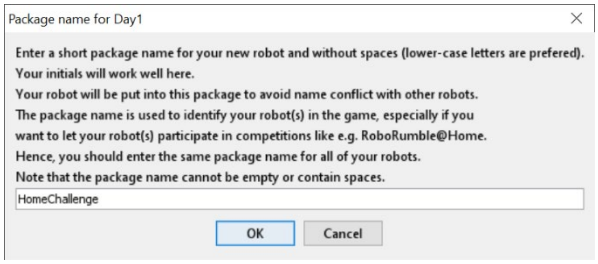
- Im Robot Editor (Source Editor) wähle **`File -> New -> JuniorRobot`**



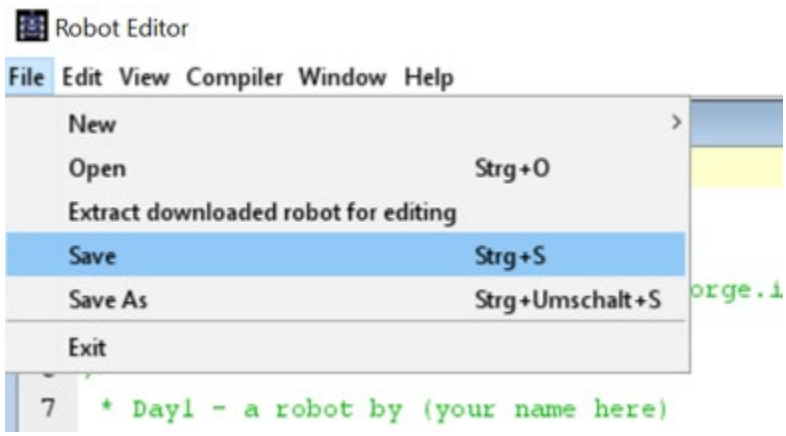
- Wir nennen den Roboter Day1



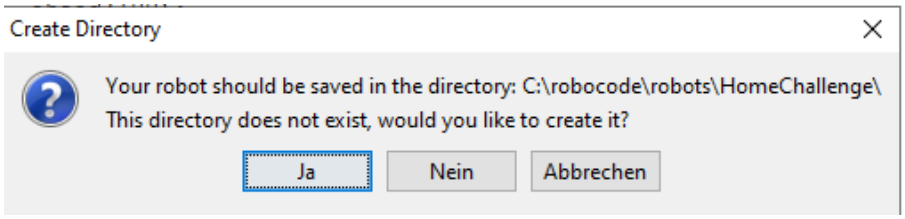
- Wenn Du nach einem Namen für ein Package gefragt wirst, nenne das Package **`HomeChallenge`**. Ein Package ist eine Sammlung in Java in der wir alle Roboter sammeln und abspeichern.



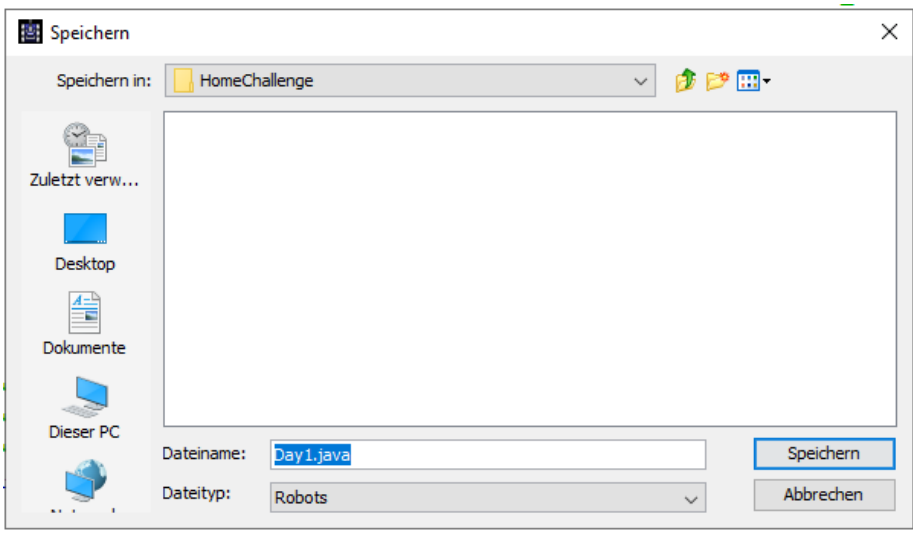
- Speichere Deinen Roboter mit dem **`File > Save`** Menu



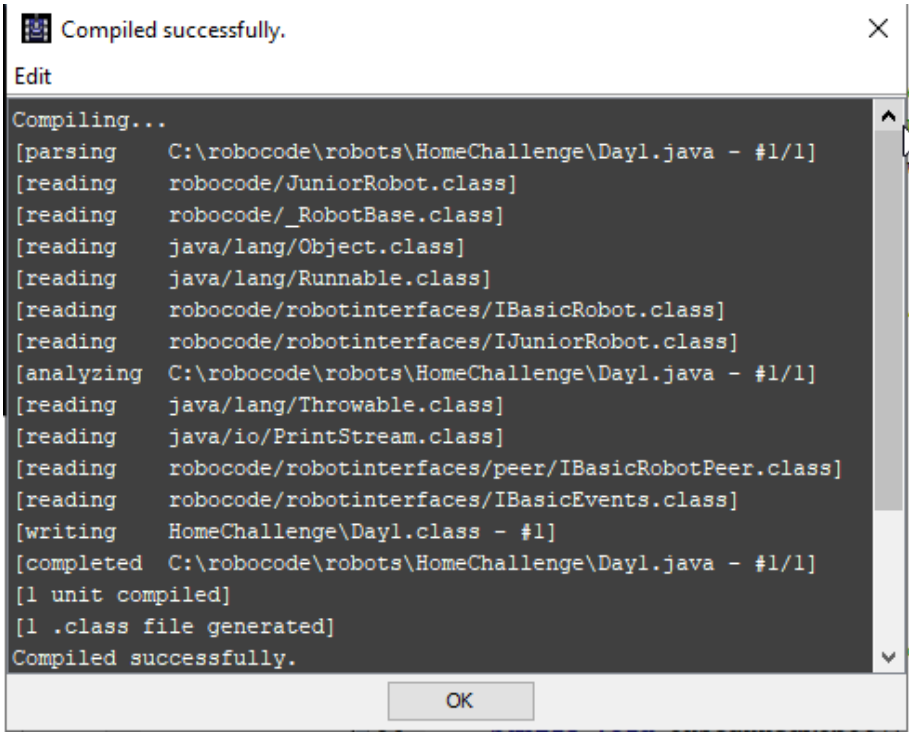
- und sage **`Ja`** um einen neuen Ordner (Directory) zu erzeugen.



- Verwende den vorgeschlagenen Namen, Day1.java, für die Datei.



- Lassen uns sofort den Roboter bauen, indem wir ihn kompilieren. Kompilieren, *compile* im Englischen, überprüft zunächst, ob der Computer alle Anweisungen versteht, und zum anderen wird der Java-Code in etwas umgewandelt, das der Computer ausführen kann.
  Klicke auf **`Compiler > Compile`**, und vergewissere Dich, dass der Code erfolgreich kompiliert wird.



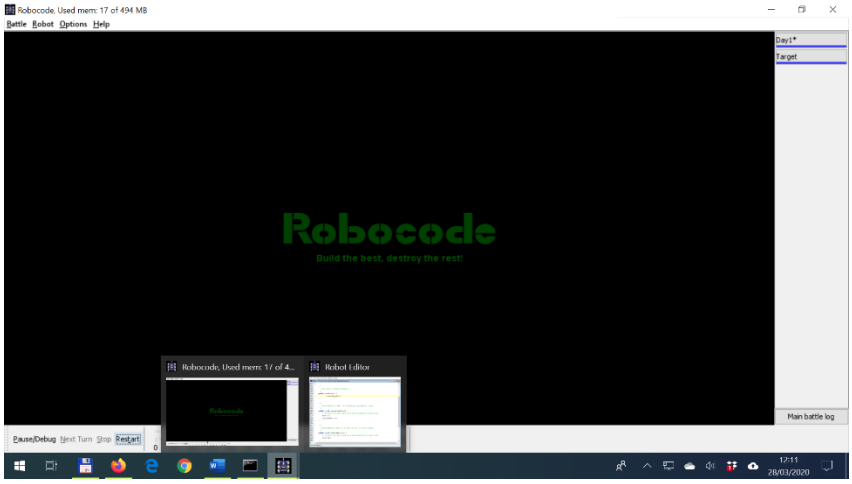
Dazu sollte das Ausgabe Fenster am Schluss die Meldung '**`Compiled successfully`**' zeigen.

Wenn Du hier irgendwelche Probleme hast; wenn Du Fehlermeldungen siehst, dann musst Du versehentlich etwas in den Editor eingegeben haben. Versuch dies zu beheben oder wenn Du den Fehler nicht findest machen Sie einen neuen Roboter von Grund auf neu; dh. starte den Abschnitt einfach nochmal von vorne.

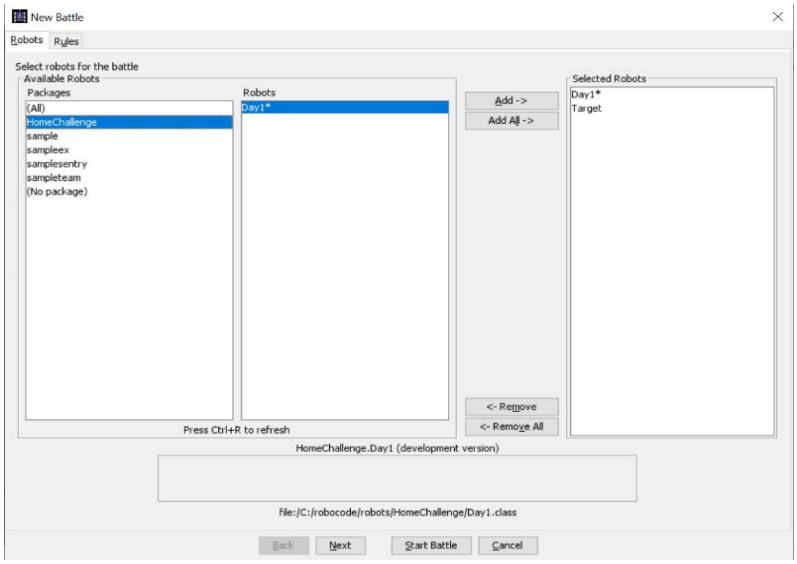
Deine Arbeit wird einfacher, wenn Du das Quellencode (Source Editor) Fenster immer offenhältst während Du mit dem Hauptprogramm weiterarbeitest.

4 Deine erste Schlacht

- Gehe nun zum schwarzen Robocode-Hauptfenster zurück, wie schon vorhergesagt lassen wir den Quellen Code Editor offen

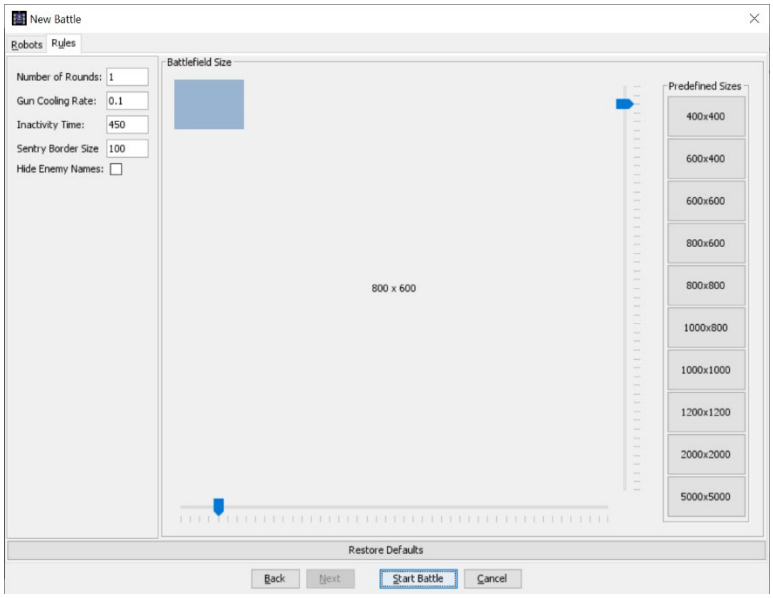


- Wähle **`Battle > New`**
- Wähle **`HomeChallenge`** aus der linken Spalte (**`Packages`**) aus, wähle dann **`Day1`** aus der mittleren **`Robots`** Spalte aus und klicke auf den **`Add`** Knopf um diesen



Roboter zur Liste der ausgewählten Roboter (**`Selected Robots`**) auf der rechten Seite des Fensters hinzuzufügen. Dann wähle aus dem **`sample`** (Beispiel) Paket den Roboter **`Target`** (Ziel) aus und füge auch den zur Liste der ausgewählten Roboter hinzu.

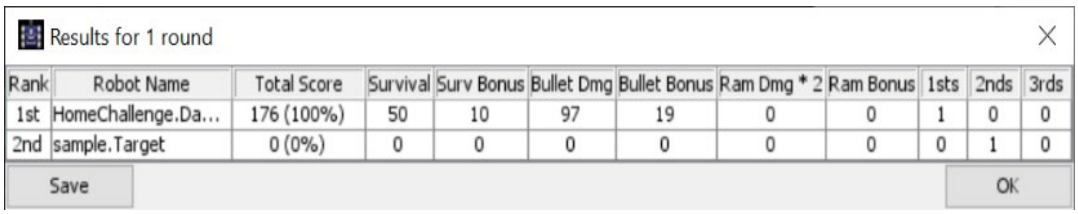
- Drücke jetzt den **`Next`** Knopf am unteren Rand des Fensters.
- Ändern im nächsten Schirm die Anzahl der Runden (**`Rounds`**) auf 1 und klicke den **`Start Battle`** Knopf.



Du siehst jetzt eine seltsame und sehr langweilige Schlacht:

- Dein Roboter bewegt sich vorwärts und rückwärts und schwingt sein Kanonenrohr am Ende jeder seiner Bewegungen rund herum
- Wenn das Geschütz auf den Feind zeigt, wird es schiessen, aber das Geschütz bewegt sich so schnell, dass es wahrscheinlich am Feind vorbeischwenkt, bevor der Schuss abgefeuert wird und das Ziel wird vollständig verfehlt.
- Jedes Mal, wenn Dein Roboter schiesst verliert er eine Einheit 1 Energie. Wenn du es schaffst, den Feind zu treffen, bekommst du etwas Energie zurück. Aber in dieser Schlacht verlieren wir nur Energie
- Jeder Roboter, auch wenn er sich nicht bewegt, beginnt Energie zu verlieren. Das heisst auch unser Ziel (Target) Roboter verliert Energie, und wird irgendwann zerstört. Man kann diese Energieeinheiten in den Einstellungen für ein Speil definieren.

Am Ende des Spiels erhältst Du Sie eine Zusammenfassung, wer gewonnen hat und wo Du Punkte erzielt hast.



Unser Roboter läuft nicht sehr gut, auch nicht gegen ein Ziel, das sich nicht viel bewegt und nicht zurückschiesst. Das wollen wir nun ändern!

Wir werden jeden Tag neue Schlachten ausführen, erinnere Dich wie man eine Schlacht (Battle) startet!

5 Erste Verbesserung

Lassen Sie uns unsere **`run()-`** Method ändern. Eine 'Method' ist ein englischer Begriff. Zu Deutsch verwendet man den Begriff Methode oder manchmal auch Funktion. In Java nennt man eine **`Method`**, ein Stück Code, der zusammengehört und dem man einen Namen gibt, den der Computer ausführen kann. In Robocode bringt eine Methode den Roboter dazu, etwas zu tun. Es gibt verschiedene Arten von Methoden, die Dein Roboter zu verschiedenen Zeiten ausführen kann, aber im Moment werden wir uns auf die **`run()-`** Methode konzentrieren. Die **`run()-`** Methode wird zu Beginn jedes Runde ausgeführt.

Robocode verwendet *Runden* (im Original **`turns`**), gemessen in Ticks, so wie eine Uhr tickt, um jeden Roboter Zeit zu geben, etwas zu tun. Ein *Tick* ist eine sehr kurze Zeit, so wird es für Dich aussehen, wie die Roboter immer etwas tun würde. Man kann sich einen turn, oder Runde, so vorstellen, dass man sagt nun bist Du dran für eine bestimmte Anzahl **`ticks`**, nun Du, nun Du. Diese Runden stellen sicher, das alle Beteiligten immer wieder Zeit erhalten etwas zu machen.

Die **`run()-`** Methode liegt zwischen den Zeilen 15 und 29 im Roboter-Editor. Die aktuelle **`run()-`** Methode führt ein paar Dinge aus:

- Es setzt die Farbe des Chassis, Kanone, Radar, Kugel und Dreh-Bogen Farbe auf orange, blau, weiss, gelb und schwarz.
- Der Roboter geht dann in eine Endlosschleife, (**`while(true)`**) in der Schleife bewegt sich der Roboter 100 Pixel vorwärts (ein Pixel ist 1 Punkt auf einem Computer-Bildschirm), dreht die Kanone nach rechts für 360 Grad (360 Grad ist ein voller Kreis), fährt rückwärts für 100 Pixel und dreht die Kanone nochmals für 360 Grad nach rechts. Es wird dies für immer und immer und immer tun (oder bis das Spiel endet).
  Hier der entsprechende Code aus dem Editor

```
while(true) {

        // Replace the next 4 lines with any behavior you would like
        // ersetzte die nächsten 4 Zeilen mit Deinen Aktionen
        ahead(100);
        turnGunRight(360);
        back(100);
        turnGunRight(360);
}
```

Im Beispiel Code, der von der Programmierumgebung gezeigt wird, sind alle Kommentare in Englisch. Wir fügen, wenn möglich eine markierte deutsche Übersetzung hinzu und werden die englische Version aber nicht löschen, damit Du Dich im Code wie vom System präsentiert zurechtfindest.

Was wollen wir also ändern?

- Wir müssen die Farben des Roboters nicht einstellen. Also löschen wir das jetzt einfach. Falls Ihr später die Farben des Roboters ändern wollt findet Ihr auf unsere Website unter **Projekte –> Code Stücke** (http://swisscoderdojo.org/de/code-snippets) eine Anleitung wie man das macht.

- Wir brauchen die Schleife nicht, da die **run()-** Methode ohnehin zu Beginn jeder Runde aufgerufen wird.
- Sich rückwärts und vorwärts zu bewegen, ist im Moment sinnlos. Niemand schiesst auf uns, also lasst uns einfach stillsitzen.
- Unsere Kanone bewegt sich zu schnell. Also, als wir den Feind entdeckt haben und eine Kugel schießen, ist das Geschütz an dem Feind vorbeigefegt und wir treffen nicht.

Ändern wir also unsere **`run()`** - Methode, dass sie wie folgt aussieht.:

```
public void run() {

	turnGunRight(1);

}
```

Wir bewegen die Waffe um 1 Grad nach rechts. Dies wird die Waffe langsam bewegen, und wenn wir einen Feind entdecken, wird es uns viel Zeit geben, einen oder mehrere Schüsse zu machen!

Lösche auch den gesamten Code innerhalb der **`OnHitWall()`**- und **`onHitByBullet()`**-**`Methode`**n. Wir bewegen uns nicht, und niemand schiesst auf uns, also brauchen wir diesen Code nicht. Die beiden Methoden sehen nun so aus:

```
public void onHitByBullet() {
	// Replace the next line with any behavior you would like
	// Ersetze die nächste Zeile mit Deinen Aktionen
}

public void onHitWall() {
	// Replace the next line with any behavior you would like
	// Ersetze die nächste Zeile mit Deinen Aktionen
}
```

Sehen wir uns also an, wie gut dieser neue Roboter funktioniert:

1. Speichern Deinen Roboter, indem Du auf **`File > Save`** klickst (oder kurz **`Strg+S`** drücken).
2. Kompiliere Deinen Roboter, indem Du **`Compiler > Compile`** wählst
3. Gehe zu deinem Robocode-Hauptfenster und beginne einen neuen Kampf mit deinem **`Day1-Roboter`** und einem **`Target-Roboter.`** Es sollte alles bereits eingerichtet sein, aber wenn nicht, kannst Du den Anweisungen von oben folgen.

Hurra! Unser Roboter hat es diesmal viel besser gemacht. Aber es bleibt heute noch genug Zeit, um eine weiter kleine Verbesserung vorzunehmen...

6 Noch eine kleine Verbesserung

Das Hauptproblem ist jetzt, dass wir nur auf den Feind schiessen, wenn unsere Kanone auf ihn zeigt, aber unsere Waffe dreht sich weiter. Wäre es nicht grossartig, wenn wir unsere Waffe auf den Feind gerichtet halten, sobald wir sie gefunden haben? Einfach!

Zu Beginn jeder Runde, wenn die **`run()`** Methode ausgeführt wird, wird das Geschütz um 1 Grad nach rechts verschoben. Sobald wir also den Feind in unseren Scannern haben, sollten wir der Waffe sagen, dass sie sich um 1 Grad nach links, also zurück wo wir auch den Feind gesehen haben, bewegen soll. Dadurch wird die Wirkung der **`Run()`** Methode aufgehoben. Wenn sich der Feind weiterbewegt und unseren Scanner verlässt, schwingt das Geschütz weiter im Uhrzeigersinn.

Kehre nun zum Roboter-Editor zurück.

Für diese Änderung müssen wir uns die **`onScannedRobot()-`** Methode ansehen. Diese Methode sollte direkt unter der **`run()-`** Methode in Zeile 21 liegen. Die **`onScannedRobot()-`** Methode wird ausgeführt (oder aufgerufen), wenn Dein Roboter einen anderen Roboter scannt (oder sieht). Im Moment hat die **`onScannedRobot()-Methode`** nur einen Kommentar und einen Befehl.

```
// Replace the next line with any behavior you would like
// Ersetze die nächste Zeile mit Deinen Aktionen
fire(1);
```

Ein Kommentar beginnt mit **//,** also zwei Mal unmittelbar hintereinander der vorwärts Schrägstrich (forward slash) und ihr könnt alles, was ihr wollt danach schreiben. Der Computer ignoriert den Rest der Zeile. Kommentare könnt ihr verwenden, um eigene Notizen zu schreiben. Diese Notizen helfen Dir Dich zu erinnern, was der Code bewirkt. Ihr könnt das Kommentar Zeichen sogar vor einen Befehl stellen und der Computer ignoriert den Befehl.

Der Befehl in der **`onScannedRobot()-`** Methode ist **`fire(1);`** . Dieser feuert das Geschütz mit einer Stärke von 1 ab. Du kannst das Geschütz mit jeder Stärke von 0,1 bis 3 in Schritten von 0,1 abfeuern. Je härter Du schiesst, dh je grösser die Zahl, desto mehr Schaden wirst Du anrichten, aber desto mehr Energie verliert Dein Roboter damit.

Fügen wir nun einen weiteren Befehl zur **`onScannedRobot()-`** Methode hinzu, die Zeile die wir mit einem Kommentar versehen haben (// <<<< Neu):

```
public void onScannedRobot() {
        // Replace the next line with any behavior you would like
        // Ersetze die nächste Zeile mit Deinen Aktionen
        fire(1);
        turnGunLeft(1); // <<<< Neu
}
```

Die neue Methode wird das Geschütz um 1 Grad nach links drehen, wodurch die Drehung der Kanone, die in der **`run()-`** Methode nach rechts gedreht wird, aufgehoben wird. Beachte auch, dass der Befehl mit einem ‘;’ (Semikolon oder Strichpunkt) endet. In Java endet jeder

Befehl mit einem `;`. Das ‘;’ sagt dem Computer, dass wir das Ende eines Befehls erreicht haben. Es ist das gleiche wie Punkt am Ende eines Satzes. Der Punkt bedeutet, dass wir das Ende dieses Satzes erreicht haben.

Jetzt speichere und kompiliere den Code wieder und starte eine neue Schlacht mit einem **`Target`** und einem **`Day1`** Roboter.

Erfolg! Du hast jetzt nun einen ziemlich effizienten Roboter, um einen Feind zu bekämpfen, der weder zurückschiesst und noch sich sehr viel bewegt.

In der nächsten Lektion werden wir die Dinge etwas schwieriger machen.

7 Bis zum nächsten Mal

Hier sind einige Dinge, die Du vor unserer nächsten Sitzung lesen kannst:

http://robowiki.net/wiki/Robocode/My_First_Robot

Falls diese Website nicht reagiert könnt ihr einen dieser alternativen Link benutzen, https://web.archive.org/web/20190826172033/http://robowiki.net/wiki/Robocode/My_First_Robot, oder http://robowiki.duckdns.org/index.php/Main_Page.

Der Obige Link enthält einige weitere Informationen über das, was wir getan haben, und kann Dir einige weitere Ideen geben, mit denen Du bis zur nächsten Lektion herumspielen kannst.