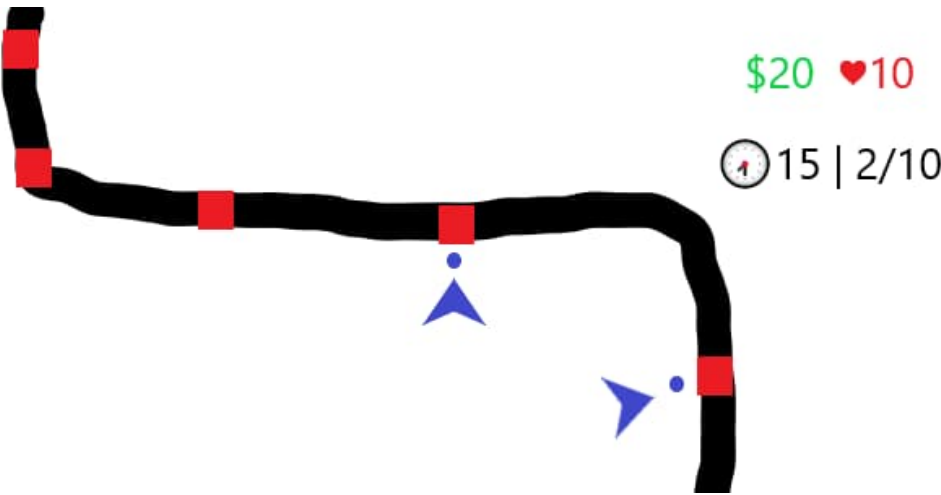


## 1 Teilnehmer/innen des Teams:

Klasse: BI19a	Team: Stählin Matteo Habtom Abrham
------------------	--

## 2 Anforderungsdefinition (Meilenstein A)

„Tower Defense“	
<b>Auftrag:</b> (Allgemeine Beschreibung)	<p><b>Nutzen: Unterhaltung</b></p> <p><b>Szenario:</b></p> <p>Tower Defense Game. Es gibt einen einfarbigen Pfad. Denn die Gegner folgen. Mann soll auf Kästchen drücken können, um Türme zu platzieren die Automatisch auf Gegner Schiessen. Man gewinnt, wen man keine Gegner durchlässt. Es gibt einen Wave timer der die Zeit zwischen den Gegner spawns anzeigt und eine Anzeige auf welcher Wave man ist. Für besiegte Gegner erhält man Geld</p> <p><b>Details:</b></p> <ul style="list-style-type: none"><li>• Pfad den Gegner folgen.</li><li>• Geld mit dem man Towers kaufen kann.</li><li>• Tower die auf Kugeln Schiessen.</li><li>• Leben die man Verliert, wen Gegner durchkommen.</li></ul> <p><b>Machbarkeitsabklärung:</b></p> <ul style="list-style-type: none"><li>• Gegner Folgen der Farbe des Pfades.</li><li>•</li></ul> 

<b>MUSS</b> <b>Kriterien:</b> (Konkrete Features, die umzusetzen sind)	<b>Folgende Features sollen implementiert werden (Funktionalität):</b> <ul style="list-style-type: none"><li>• Geld mit dem man Towers kaufen kann.</li><li>• Gegner die einem Pfad folgen.</li><li>• Mehrere Levels.</li><li>• Leben, die Man verliert falls ein Gegner den Pfad beendet.</li><li>• Towers die auf Gegner schiessen.</li></ul>
<b>KANN</b> <b>Kriterien:</b> (Konkrete Features, die optional sind)	<b>Folgende Features können zusätzlich implementiert werden: (Kreativität)</b> <ul style="list-style-type: none"><li>• Mehrere Gegnerarten (Schnellere oder solche mit mehr Leben).</li><li>• Mehrere Tower arten.</li><li>• Magie mit Abklingzeit die man irgendwo einsetzen kann.</li></ul>

## 2.1 Planung LB2

<i><b>MS</b></i>	<i><b>Tätigkeit / Abgabe</b></i>	<i><b>Soll-Datum</b></i>	<i><b>Ist-Datum</b></i>
A	<b>Projektstart</b> <ul style="list-style-type: none"><li>➤ Team Bildung</li><li>➤ <b>Wahl / Ausarbeitung der Anforderungsdefinition</b></li></ul> Abnahme Anforderungsdefinition durch Lehrperson		
B	<b>Teamaufgabe 1:</b> <ul style="list-style-type: none"><li>➤ <b>Abgabe: Lösungsdesign</b> (Analyse, Design: Funktionsmodell, UseCase, GUI, Storyboard)</li></ul>		
B2	<b>Teamaufgabe 2:</b> <ul style="list-style-type: none"><li>➤ Abgabe: Testvorschrift und Testfälle</li></ul>		
C	<b>Einzelaufgabe 3:</b> <ul style="list-style-type: none"><li>➤ Abgabe Szenario (.zip) mit Inline-Dokumentation, Systemdokumentation (UML Klassen-, Sequenzdiagramm)</li><li>➤ <b>Fachgespräch Projektabnahme</b></li></ul>		
C2	<b>Einzelaufgabe 4:</b> <ul style="list-style-type: none"><li>➤ Abgabe: Ausgefüllter Systemtest</li></ul>		

### 3 Lösungsdesign (Meilenstein B: Teamaufgabe 1)

Anhand der Analyse wurde folgendes Lösungsdesign entworfen:

### 3.1 Funktionsmodell

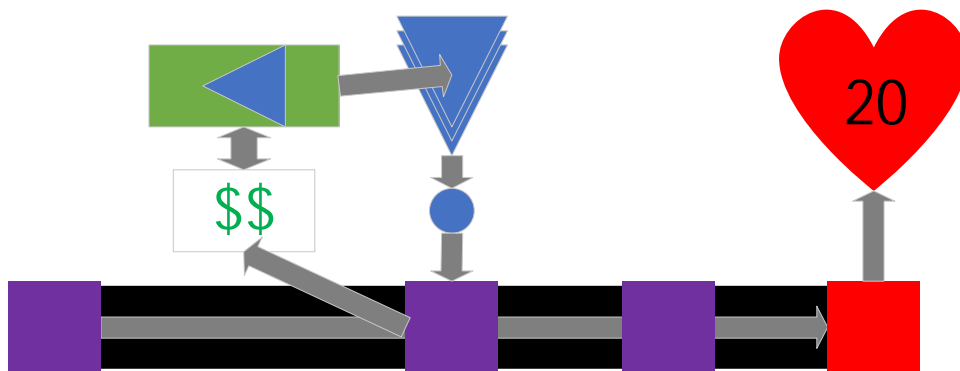
Im Folgenden sind die erwarteten Eingaben und Ausgaben beschrieben / dargestellt:

Objekte:

Tower, Bullet, Enemy, Goal, Button\_Buy, Geld

Konzepte:

Gegnerauswahl, Pfadfolgen, Schiessen, Goal erreicht, Tower kaufen,

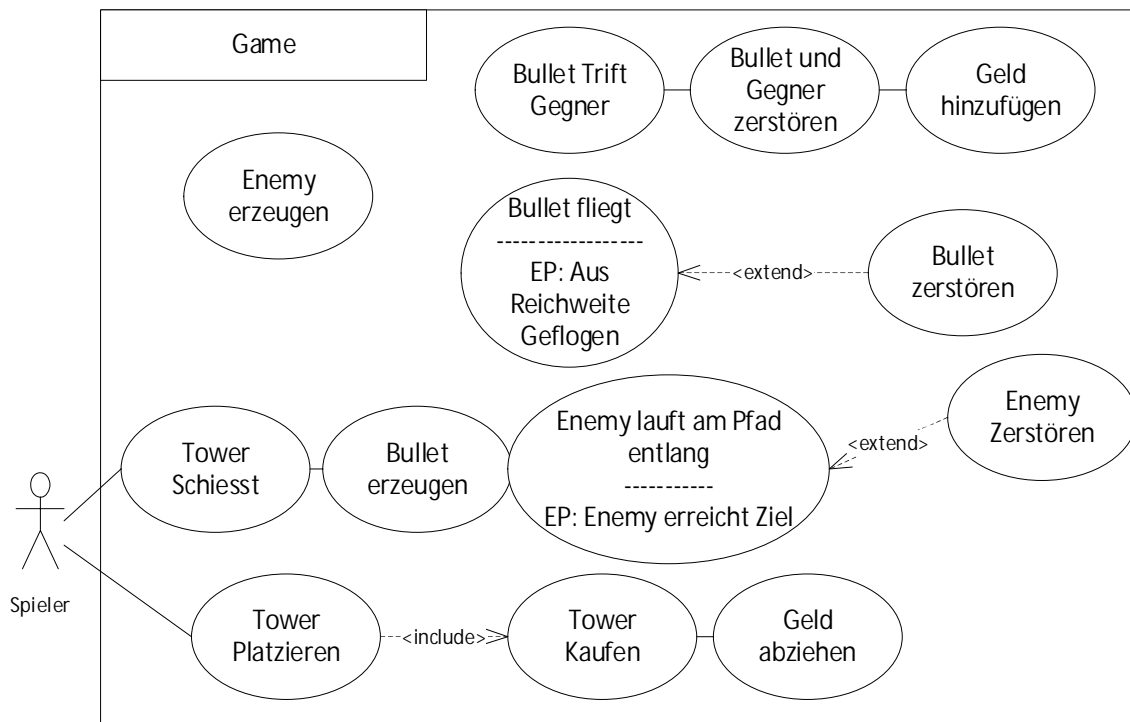


Legende:

- Enemy folgt dem Pfad zum Goal
- Button\_Buy kauft Tower mit Geld und platziert ihn
- Tower sieht Enemy innerhalb Reichweite
- Tower schießt auf Gegner (Manuel oder Automatisch)
- Bullet fliegt zum Enemy
- Bullet zerstört Enemy
- Zerstörter Enemy erzeugt Geld
- Enemy der zum Goal kommt nimmt Leben weg

### 3.2 Anwendungsfälle (UseCases)

Folgende Anwendungsfälle sind hier detailliert dokumentiert:



#### Legende:

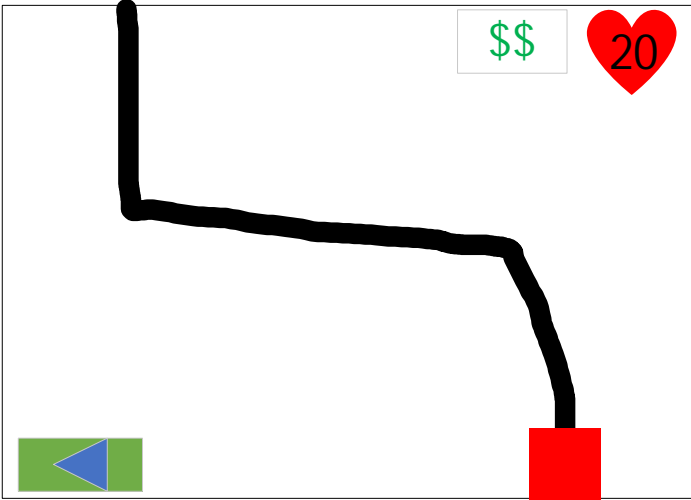
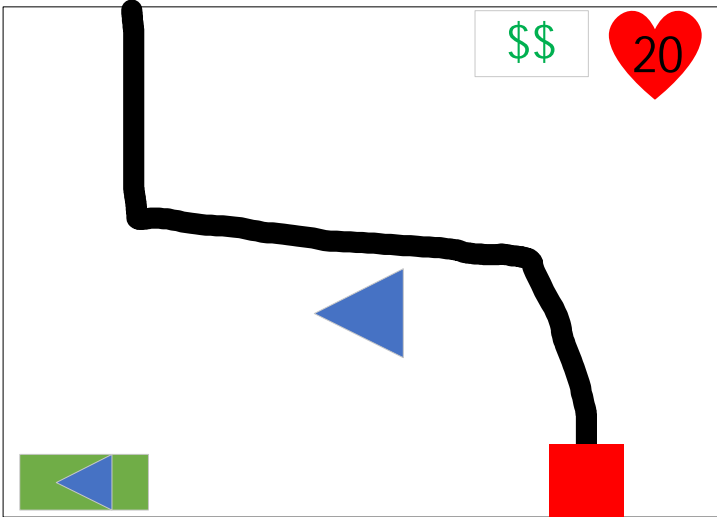
- Sobald der Turm (Tower) gekauft und platziert ist, wird das Geld abgezogene
- Enemy (Gegner) wird automatisch erzeugt
- Enemy (Gegner) läuft am Pfad und wird zerstört, sobald er das Ziel erreicht hat.
- Sobald das Bullet ausser Reichweite geflogen ist, wird er zerstört
- Der Turm (Tower) schiesst und Kugel (Bullet) wird erzeugt
- Die Kugel (Bullet) trifft Gegner (Enemy) beide wird zerstört und Geld wird hineingefügt

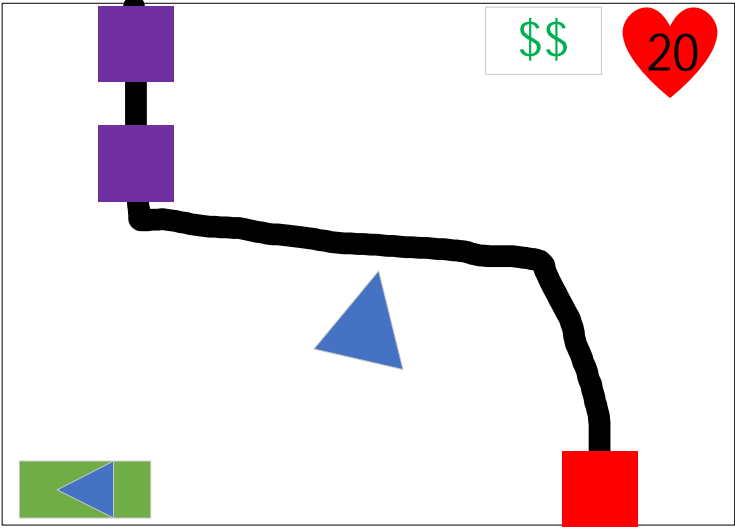
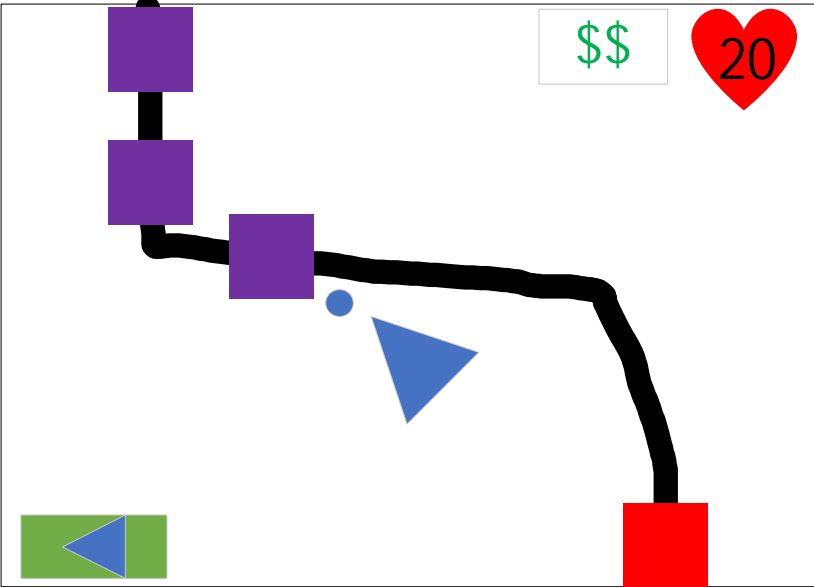
### 3.3 Ablauf

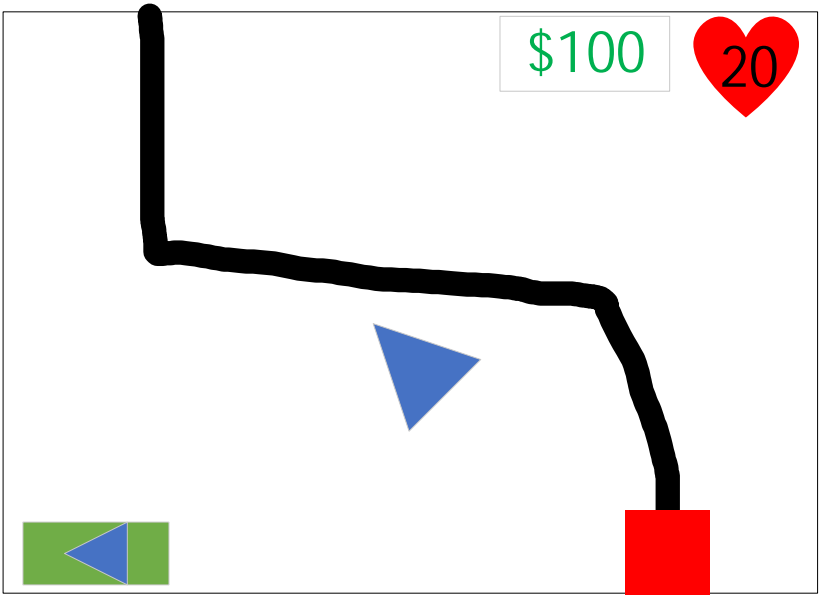
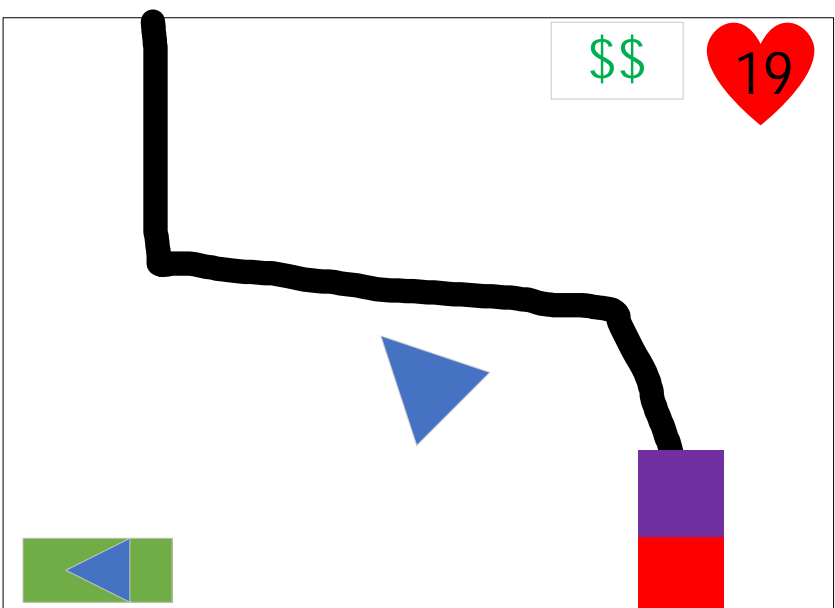
Aus Benutzersicht ist folgender Ablauf des Programms zu erwarten:

...

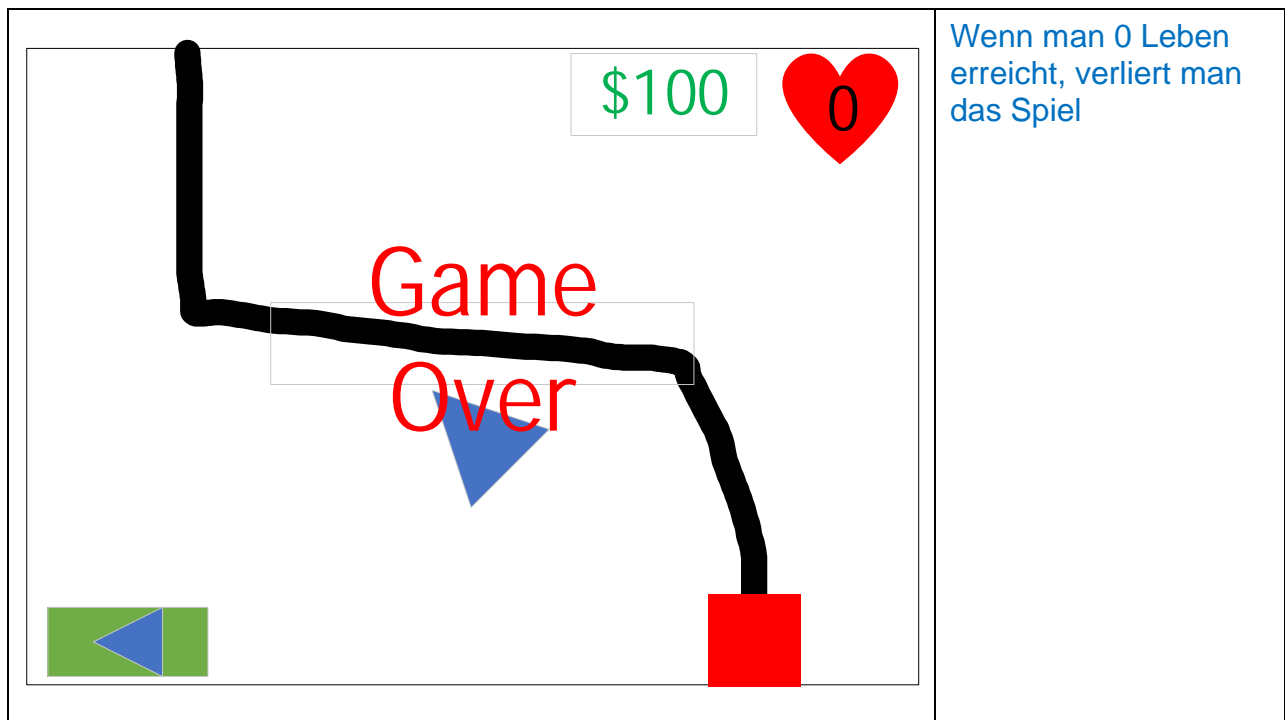
(Storyboard)

	<p><b>Startsituation</b></p> <p>Szenario muss gestartet werden Run&gt;</p> <p>Die Mappe wird eingerichtet mit Goal, Buy Buttons, Geld und Leben.</p>
	<p><b>Useraktivität</b></p> <p>Tower kann mit genügend Geld gekauft werden.</p>

	<p>Enemys spawnen nach einer Zeit und Folgen dem Pfad zum Ziel.</p>
	<p>Useraktivität Entweder Tower oder Spieler schiesst an die Enemy zu.</p>

 <p>The diagram shows a game state. A green base with a blue triangle is at the bottom left. A red square target is at the bottom right. A black path starts from the top left, goes down, then right, then down again to the red target. A blue triangle is on the horizontal segment of the path. In the top right corner, there is a green box with '\$100' and a red heart with '20'.</p>	<p>Getötete Gegner geben Geld.</p>
 <p>The diagram shows a game state. A green base with a blue triangle is at the bottom left. A purple and red square target is at the bottom right. A black path starts from the top left, goes down, then right, then down again to the target. A blue triangle is on the horizontal segment of the path. In the top right corner, there is a green box with '\$\$' and a red heart with '19'.</p>	<p>Ein Gegner, der das Ziel erreicht zieht leben ab.</p>





#### 4 Testvorschrift (LB2 Meilenstein B2: Teamaufgabe 2)

Testbeschreibung und vorbereitetes Testprotokoll siehe Dokument  
[M226B\\_LB2\\_Testvorschrift\\_MS-B2.docx](#)

#### 5 Systemdokumentation (Meilenstein C: individuelle Aufgabe 3)

Das erstellte Java-Projekt (Greenfoor-Szenario) ist hier detailliert abgelegt:

[M226B\\_Aufgabe\\_3\\_Szenario\\_IhrName.zip](#)

##### 5.1 Statisches Design: Klassendiagramm

Folgend die statische Struktur des Szenarios

...

(UML Klassendiagramm mit Assoziationen und Kardinalitäten)

##### 5.2 Umfang / Abgrenzung / Änderungen gegenüber Design

Aufgrund unten beschriebener Umstände sind Anpassungen des ursprünglichen Lösungsdesigns gemacht worden:

...

(Umstände / Anpassungen / Veränderungen)

### 5.3 Funktionalität der Implementation.

Zusätzlich zu der Inline-Dokumentation sind hier folgende Funktionen detailliert beschrieben:

...

(Ausführliche Beschreibung der internen Funktionen  
oder Verweis zum Inline-Kommentar mit JavaDoc! (`/** @param @return **`))

### 5.4 Dynamische Struktur: Sequenzdiagramm

Ein zentraler Ablauf eines UseCases ist im Folgenden dargestellt:

...

(Darstellung eines zentralen Ablaufs mittels Sequenzdiagramm)

**Trace: ...**

...

## 6 Bedienungsanleitung (Meilenstein C: individuelle Aufgabe 3)

...

## 7 Testprotokoll (LB2 Meilenstein C2: individuelle Aufgabe 4)

Ausgefülltes Testprotokoll siehe Dokument  
[M226B\\_LB2\\_Testvorschrift\\_MS-C2\\_Name.docx](#)