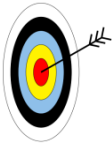






2a AngryDot

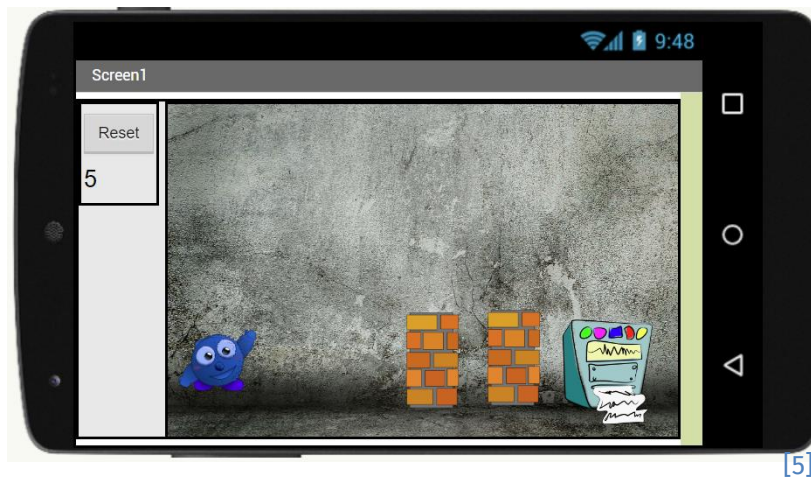


Ihr habt euch für **AngryDot** entschieden: ein lustiges Spiel, bei dem es darum geht, den

InfoDot (kurz: Dot)  [1] zu werfen, um den **Supercomputer**  [2] zu zerstören. Dieses Arbeitsblatt wird euch helfen, eine App zu erstellen, die...

- × einen Dot auf einem **Spielfeld**  [3] erscheinen lässt.
- × es dem Spieler bzw. der Spielerin ermöglicht, den Dot durch **Schleudern** zu werfen.
- × die Verteidigung  [4] des Supercomputers, bzw. den Supercomputer selbst, zerstört, wenn diese getroffen werden.




Schwierigkeitsgrad: ★★★★★



[5]



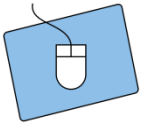
Während des gesamten Moduls geben euch die Arbeitsblätter Hinweise zur Umsetzung. Achtet dabei einfach auf die folgenden Symbole, die...

- × euer Arbeiten strukturieren und Teilziele aufzeigen, 
- × euch Hilfen geben, Wichtiges, Schwieriges, etc. kennzeichnen und 
- × die Arbeitsaufträge und Aktionen beinhalten. 

2a AngryDot

Anlegen des Projekts

Wie bei jeder neuen App müsst ihr zunächst ein neues Projekt anlegen.



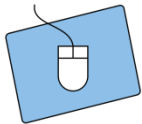
1. Erstellt euer Projekt, und gebt ihm einen passenden Namen.
2. Verbindet den App Inventor wie auf dem ersten Arbeitsblatt beschrieben mit dem Tablet/Smartphone.

Der Aufbau eurer App

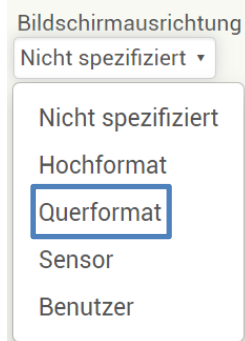
Als Nächstes müsst ihr euch Gedanken darüber machen, wie eure App aussehen soll. Eure App braucht Platz für folgende Dinge (die ihr der Reihe nach einbauen werdet):

- eine **Spielfläche**, auf der der Dot sich bewegen kann,
- eine **Reset-Taste**, mit der ihr das Spiel neustartet,
- eine **Lebansanzeige**, die nach jedem Versuch ein Leben herunterzählt.

Das sind schon recht viele Sachen. Damit diese alle Platz haben, solltet ihr zunächst die **Bildschirmausrichtung** (Screen1) von **Nicht spezifiziert** in **Querformat** ändern.



1. Wählt unter den **Komponenten Screen1** aus.
2. Sucht unter den **Eigenschaften** den Eintrag **Bildschirmausrichtung**.
3. Ändert den Wert von **Nicht spezifiziert** in **Querformat**.



[6]

Die Bestandteile eurer App

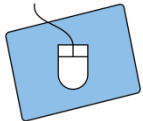
Nun könnt ihr anfangen, die einzelnen Elemente eurer App hinzuzufügen und diese mit Hilfe der vorgestellten **Anordnungen** sinnvoll zu platzieren.

- Für die **Spielfläche** braucht ihr eine **Zeichenfläche**; diese findet ihr unter **Zeichnen und Animation**.

2a AngryDot

Eine Zeichenfläche könnt ihr euch wie eine Leinwand vorstellen, auf der sich Objekte wie der Dot bewegen können.

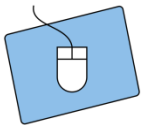
- Für die **Reset-Taste** braucht ihr eine **Taste** mit dem Text „Reset“.
- Für die **Lebensanzeige** benötigt ihr eine **Bezeichnung** mit dem Text „5“ bzw. der Anzahl an Leben, die euer Dot haben soll.



1. Macht euch Gedanken darüber, wie ihr die Elemente positionieren wollt.
2. Zieht die **Anordnungen**, die ihr für euer Layout braucht, in den Betrachter hinein.
3. Fügt eurer App die Bestandteile (Zeichenfläche, Taste, Bezeichnung) hinzu.

Der erste Test

Nachdem ihr die Bestandteile sinnvoll angeordnet habt, könnt ihr direkt testen, wie das Ganze auf dem Tablet/Smartphone aussieht.



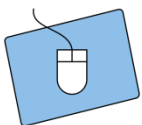
1. Startet dazu die App auf dem Tablet/Smartphone, wenn ihr dies noch nicht gemacht habt.
2. Gefällt ihr die Anordnung? Wenn ja, dann könnt ihr weitermachen. Ansonsten überarbeitet sie einfach nochmal.



Lass eure App auf dem Tablet/Smartphone laufen. Der App Inventor aktualisiert alle Änderungen, die ihr vornehmt, und zeigt euch diese direkt an, ohne dass ihr die App neustarten müsst. Sollte dies einmal nicht der Fall sein, dann setzt die Verbindung zurück.

Umbenennen nicht vergessen 😊

Da ihr jetzt die ersten Bestandteile in eurer App integriert habt, ist es an der Zeit, diese mit sinnvollen Namen zu versehen, damit ihr sie im Blöcke-Editor besser unterscheiden könnt.



Gebt den Bestandteilen eurer App sinnvolle Namen.

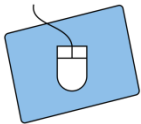
2a AngryDot

Ein Hintergrund für eure Spielfläche

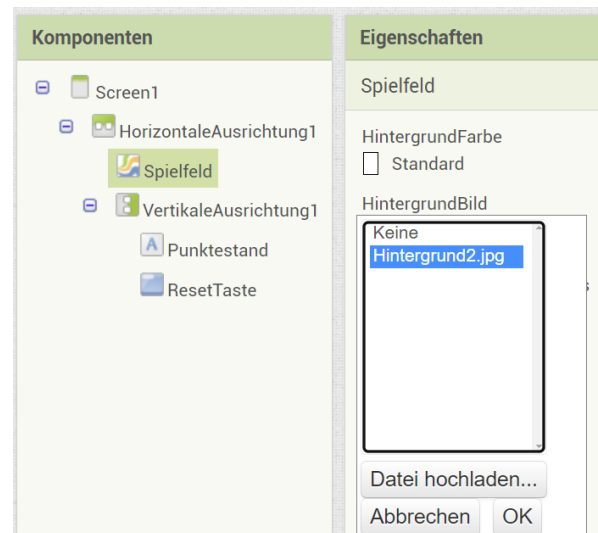
Damit der Dot sich auf der Spielfläche wohlfühlt, soll diese einen Hintergrund bekommen. Ladet dazu einen der beiden Hintergründe aus dem folgenden Ordner hoch:

Desktop / InfoSphere goes Android / AngryDot

Im Anschluss müsst ihr eurer Zeichenfläche den Hintergrund noch zuweisen:



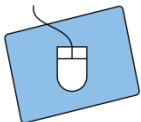
1. Wählt unter den **Komponenten** die **Zeichenfläche** aus.
2. Klickt unter den **Eigenschaften** auf **HintergrundBild**, und weist der Zeichenfläche einen der beiden Hintergründe zu.



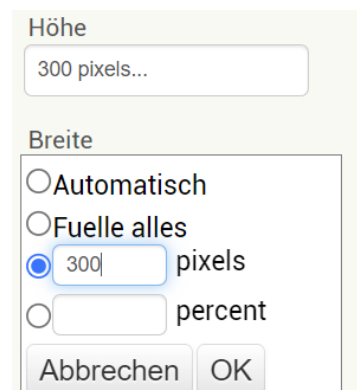
[7]



Wie ihr schon von Blatt 1 wisst, muss man die Größe beim App Inventor manchmal per Hand einstellen.



3. Ihr habt immer noch eure **Zeichenfläche** (die in der Abbildung schon in Spielfeld umbenannt ist) ausgewählt. Sucht in den **Eigenschaften** nach **Breite** und **Höhe**, und setzt beide auf 300 Pixel.
4. Testet jetzt eure App, und ändert gegebenenfalls die Anzahl der Pixel, damit es auf dem Tablet/Smartphone gut aussieht.



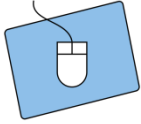
[8]

Auf der nächsten Seite geht es weiter mit dem Dot.

2a AngryDot

Der Dot

Als Nächstes soll der Dot auf das Spielfeld.

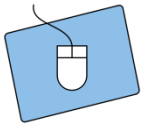


1. Wechselt in der Palette zu **Zeichnen und Animation**.
2. Zieht eine **ZeichenAnimation** direkt in den Hintergrund.
3. Ändert den Namen der ZeichenAnimation in Dot.
4. Ladet nun aus dem Ordner, aus dem ihr auch den Hintergrund habt (S. 4), den Dot hoch, und weist ihn der ZeichenAnimation zu.



Eine **ZeichenAnimation** ist ein besonderes Bild. Im Gegensatz zu einem normalen Bild kann die ZeichenAnimation sich auf dem Spielfeld bewegen.

Außerdem hat euer Dot eine **Position** auf dem Spielfeld. Diese könnt ihr in den Eigenschaften als **X- und Y-Koordinaten** ablesen bzw. verändern.

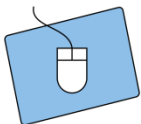


1. Wählt den Dot unter den Komponenten aus.
2. Ändert unter den Eigenschaften die X- und Y-Koordinaten, und sucht euch eine schöne Startposition für den Dot aus.

Der Supercomputer und die Wände

Neben dem Dot braucht ihr noch weitere **ZeichenAnimationen**:

- eine ZeichenAnimation für den **Supercomputer** und
- zwei ZeichenAnimationen für die **Wände**.



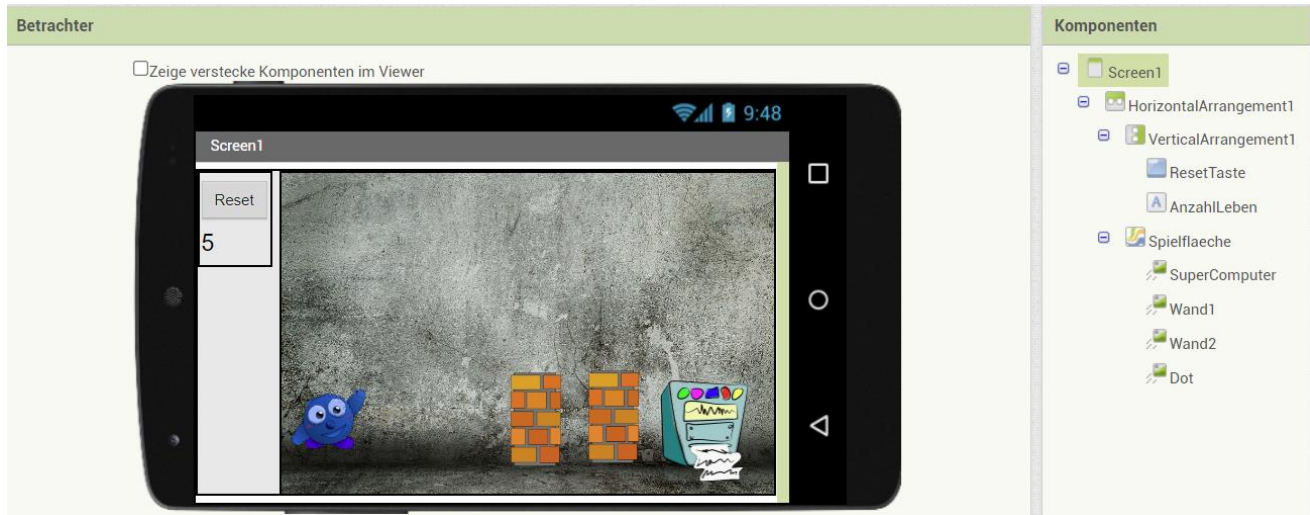
1. Verfährt genau wie im vorherigen Abschnitt, und erstellt die ZeichenAnimationen an einer sinnvollen Position.
2. Testet das Gesamtbild eurer App.

Hinweis: Wenn ihr die Wände drehen möchtet, müsst ihr unter den Eigenschaften den Wert der **Richtung** ändern.

2a AngryDot

Ein Zwischenfazit

So oder so ähnlich sollte eure App jetzt aussehen. Falls ihr noch Fragen habt, spricht mit einem Betreuer oder einer Betreuerin.



[9]

Die Bewegung des Dots

Als Erstes soll der Dot durch eine Wischbewegung losfliegen.



1. Wechselt in den Blöcke-Editor.
2. Wählt bei eurem Dot den Block **wenn-Dot.Geschleudert** aus.

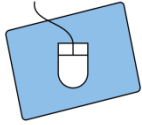
Wenn der Dot geschleudert wird, dann wird etwas gemacht.

```
wenn Dot .Geschleudert
  x y Geschwindigkeit Ueberschrift xvel yvel
  mache
```

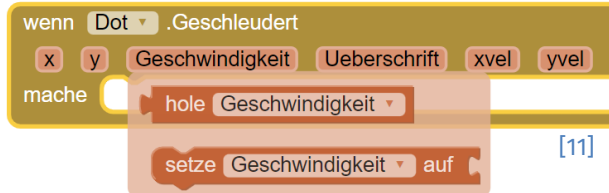
[10]

Auf der nächsten Seite geht es weiter.

2a AngryDot



3. Außerdem werden durch diese Funktion noch Variablen erzeugt, die für euch wichtig sind: die **Geschwindigkeit** und die **Richtung**. Diese könnt ihr erhalten, wenn ihr den Mauszeiger über den Namen der Variablen bewegt.



Hinweis: Die Ursprungssprache des App Inventors ist Englisch. Aufgrund eines Übersetzungsfehlers wurde **heading** nicht als **Richtung**, sondern als **Ueberschrift** übersetzt.

4. Jetzt müsst ihr dem Dot noch die neuen Werte für die **Geschwindigkeit** und die **Richtung** zuweisen.



Wenn ihr den Pfeil anklickt könnt ihr zwischen den Variablen (Geschwindigkeit, Richtung) wählen.

5. Versucht, die Blöcke selbstständig zu kombinieren, und fragt eine*n Betreuer*in, falls ihr Hilfe braucht.
6. Testet die Schleuderfunktion.

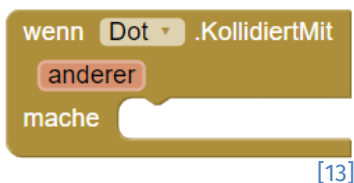
Auswertung nach jedem Versuch

Der Wurf des Dots ist beendet, wenn...

- er mit einer **Wand** kollidiert,
- er mit dem **Supercomputer** kollidiert oder
- er den **Rand des Spielfeldes** berührt.

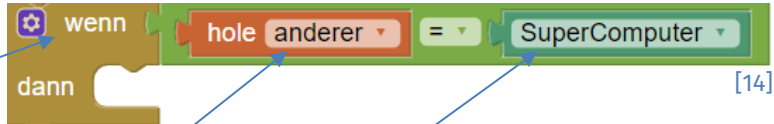
Nach jedem Wurf müssen diese Fälle überprüft werden, und anschließend muss entschieden werden, ob weitergespielt oder das Spiel beendet wird. Diese Fälle werdet ihr in den folgenden Abschnitten einbauen. Für die ersten beiden Fälle (Wand und Supercomputer) könnt ihr die Funktion **wenn-Dot.KollidiertMit** benutzen.

Wenn der Dot mit etwas **kollidiert, dann** kann etwas gemacht werden.

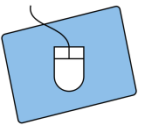


2a AngryDot

Zusätzlich erhaltet ihr die Komponente, die mit der Kollision stattgefunden hat. **Anderer** kann also eine Wand oder der Supercomputer sein. Den Block zum Vergleichen zweier Objekte findet ihr unter Logik. Im nächsten Schritt sollt ihr kontrollieren, welcher der drei Fälle eingetroffen ist. Ob der Supercomputer getroffen wurde, könnt ihr folgendermaßen überprüfen:



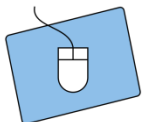
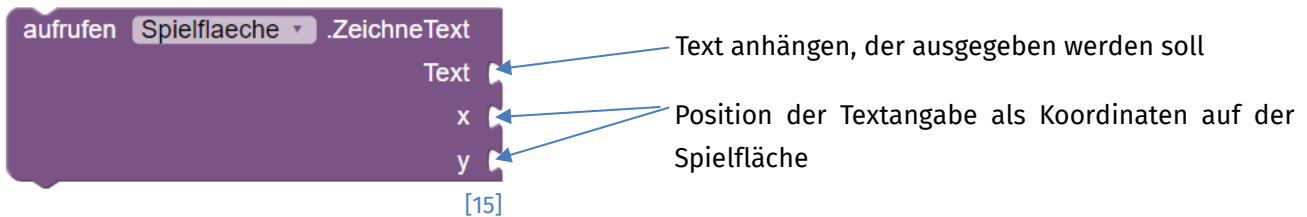
Wenn die getroffene Komponente (**andere**) der **Supercomputer** ist, dann kann etwas gemacht werden.



Erstellt das Wenn-dann-Konstrukt für die folgenden Fälle: Supercomputer und Wände. Zögert nicht zu fragen, falls ihr Hilfe braucht.

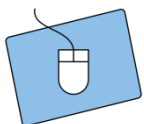
Der Supercomputer wurde getroffen

Wenn der Supercomputer getroffen wurde, müsst ihr dem Spieler oder der Spielerin sagen, dass er oder sie gewonnen hat. Dazu könnt ihr einen Text mit der folgenden Konstruktion auf dem Bildschirm ausgeben:



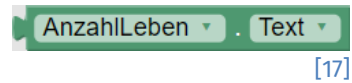
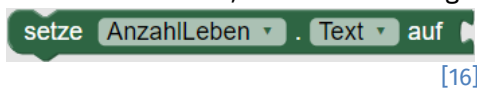
1. Baut den Text, dass das Spiel gewonnen wurde, ein.
2. Haltet außerdem den Dot an, indem ihr dessen Geschwindigkeit auf 0 setzt.

Eine Wand wurde getroffen

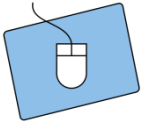


Für getroffene Wände müsst ihr die folgenden Schritte umsetzen:

1. Zieht ein Leben ab, indem ihr die folgenden Blöcke nutzt:



2a AngryDot



2. **Zerstört** die getroffene **Wand**, indem ihr sie unsichtbar macht:

```
setze Wand1 . Aktiviert auf falsch [20] [21]
wahr [22]
```

Indem ihr den kleinen **Pfeil** anklickt, öffnet ihr eine Liste, in der ihr u. a. den Punkt **Sichtbar** findet.

3. Setzt den Dot an seine **Startposition** zurück.

```
aufrufen Dot .BewegeZu [23]
x
y
```

4. **Haltet** den Dot **an**, indem ihr seine Geschwindigkeit zurücksetzt.

```
setze Dot . Richtung auf 0 [24] [25]
```

Geschwindigkeit auswählen

5. Überprüft, ob der Spieler oder die Spielerin **verloren** hat (Leben = 0). **Wenn** das Leben gleich 0 ist, **dann** gebe einen Text aus, der dem Spieler bzw. der Spielerin sagt, dass er/sie verloren hat.
6. Ihr habt sowohl die Treffer für den Supercomputer als auch für die Wände implementiert? Dann testet eure App.

Super! Bis hier ist es schon ein ganzes Stück Arbeit gewesen. Nun kommt der Endspurt.

Der Rand wurde getroffen

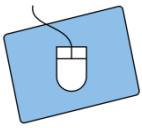
Jetzt fehlt euch noch der Rand eurer Spielfläche.



1. Benutzt die folgende Funktion, um zu überprüfen, ob der Dot mit dem Rand der Spielfläche kollidiert ist:

```
wenn Dot .RandErreicht [26]
  Rand
  mache
```

2a AngryDot

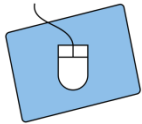


2. Wenn der Rand getroffen wurde, müsst ihr Folgendes machen:

- den Dot auf die **Startposition** zurücksetzen,
- ein **Leben abziehen**,
- überprüfen, ob ihr noch **genügend Leben habt** und eventuell ausgeben, dass der Spieler oder die Spielerin **verloren hat**.

Die Reset-Taste

Als Letztes müsst ihr dafür sorgen, dass das Spiel zurückgesetzt werden kann. Dazu braucht ihr die Reset-Taste.



1. Wenn die **Reset-Taste** gedrückt wird, dann sollen die **Grundeinstellungen** wiederhergestellt werden. Verwendet deshalb zunächst folgende Funktion:

```
wenn ResetTaste .Klick
mache
```

[27]

2. Überlegt euch nun, was die **Grundeinstellungen** sind und dementsprechend in den **mache**-Teil gehört. Hierbei können euch die folgenden **Hinweise** helfen:

- Die **Lebensanzeige** hat eine Grundeinstellung.
- Die **Position** des Dots muss auf den Startwert zurückgesetzt werden.
- Die **Geschwindigkeit** des Dots muss zurückgesetzt werden.
- **Zerstörte Komponenten** (Wände, Supercomputer) müssen wieder **sichtbar** gemacht werden.
- Die **Meldung**, dass der Spieler oder die Spielerin verloren hat, muss **deaktiviert** werden.



Mit `aufrufen Spielflaeche .Lösche` [28] kann ein Text auf der Spielfläche gelöscht werden.

2a AngryDot



Gratulation 😊

*Damit habt ihr ein voll funktionsfähiges AngryDot-Spiel programmiert. Im Folgenden findet ihr noch Tipps, Hinweise und Anregungen für Erweiterungen. Falls ihr lieber ein neues Spiel programmieren möchtet, dann wendet euch an die Betreuer*innen.*

Erweiterungen

Hier findet ihr einige Ideen, um das Spiel noch zu erweitern.

Mehrere Wände

Ihr könnt mehrere Wände einbauen, die unterschiedliche Positionen haben. Zusätzlich kann die Anzahl der Leben erhöht werden, damit es nicht zu schwierig wird.

Bewegte Wände

Ihr könnt programmieren, dass die Wände sich bewegen. Dazu müsst ihr euch überlegen, in welchem Bereich sich die Wände bewegen sollen und wann sie die Richtung ändern. Beispielsweise könntet ihr die Wände immer bis zum Rand laufen lassen. Wenn dann eine Kollision der Wände mit dem Rand stattfindet, ändert ihr die Bewegungsrichtung um 180°.

Extra-Leben

Ihr könnt einen zweiten Dot auf dem Spielfeld einfügen. Wenn man diesen trifft, bekommt man ein Extra-Leben geschenkt.

Quellenverzeichnis:





Abb. 1, 4, , , ,  – Quelle: InfoSphere, CC BY-SA 4.0 Attribution-ShareAlike 4.0 International (<https://creativecommons.org/licenses/by-sa/4.0/>).

Abb. 2 – Quelle: openclipart, Autor: mi_brami (<https://openclipart.org/download/168084/old-computer.svg>), CC0 1.0 (<https://creativecommons.org/publicdomain/zero/1.0/>), abgerufen am: 06.03.2023.

Abb. 3 – Quelle: pxhere (<https://pxhere.com/de/photo/972975>), CC0 1.0 (<https://creativecommons.org/publicdomain/zero/1.0/>), abgerufen am: 06.03.2023.

Abb. 5 bis 28 – Quelle: Screenshot des MIT App Inventor 2 (<http://ai2.appinventor.mit.edu/?locale=de>), CC BY-SA 4.0 (<https://creativecommons.org/licenses/by-sa/4.0/>), erstellt am: 06.03.2023.