

Aktivitäten im Zustandsdiagramm

Do, Entry, Exit, Transition
am Beispiel eines Schulbusses



Ich bin Mik, Dein Mikrocontroller



Aktivitäten im Zustandsdiagramm



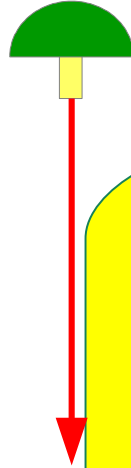
Haltestelle



Ein Schulbus soll automatisiert
werden



Aktivitäten im Zustandsdiagramm



Haltestelle

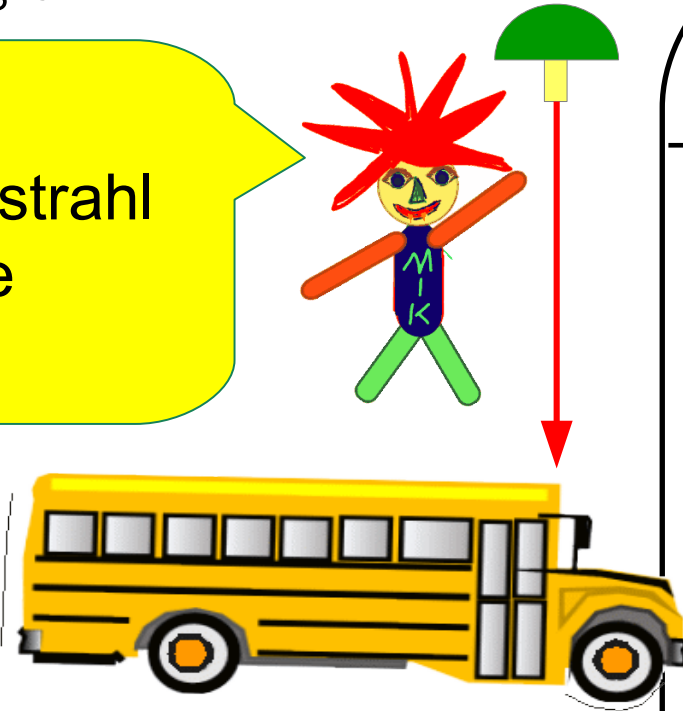


Wenn der Schulbus die Haltestelle erreicht, wird ein Countdown-Timer gestartet. So lange haben die Fahrgäste Zeit zum ein- und aussteigen. Danach fährt der Bus wieder ab,



Aktivitäten im Zustandsdiagramm

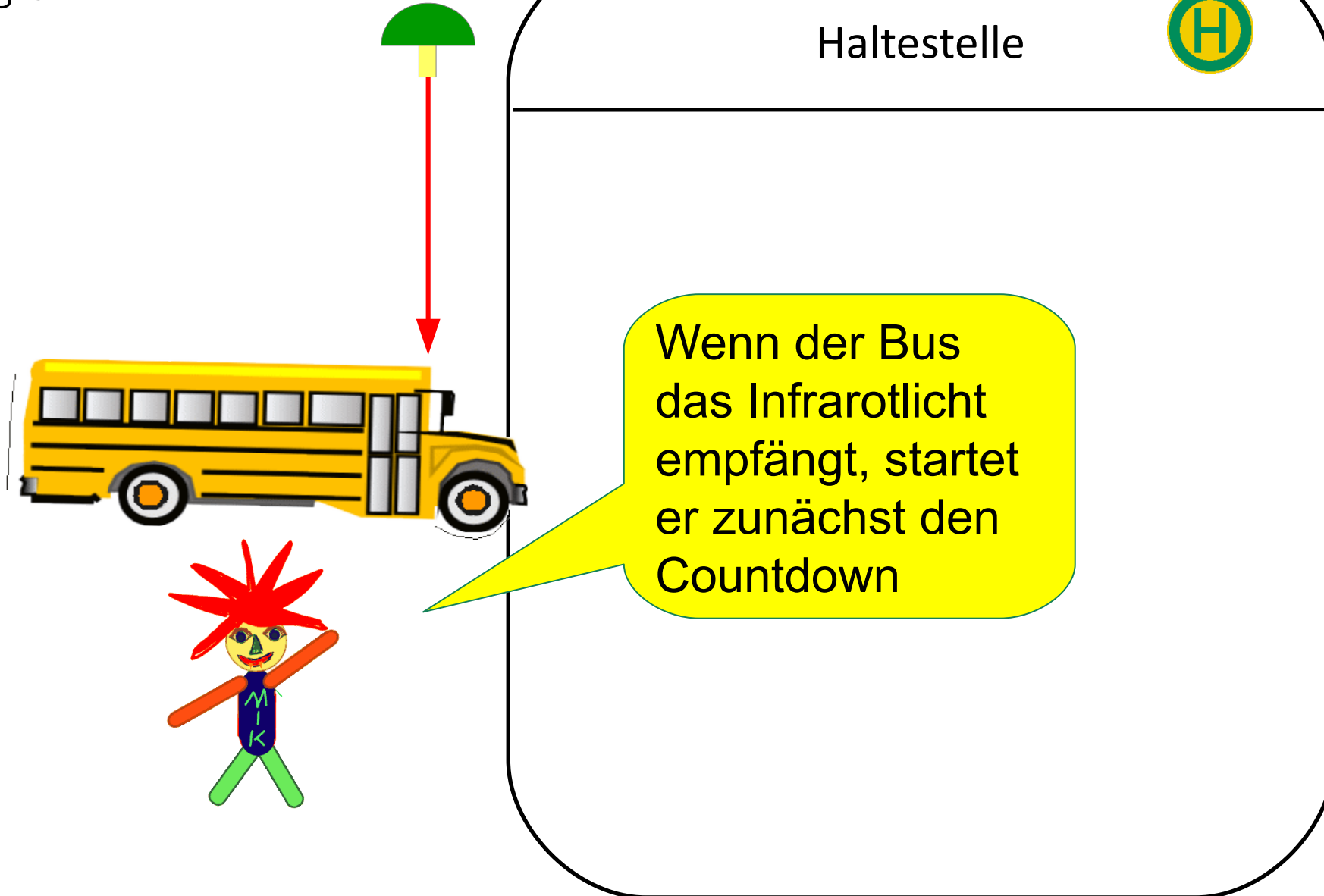
Ein
Infrarotlichtstrahl
markiert die
Haltestelle.



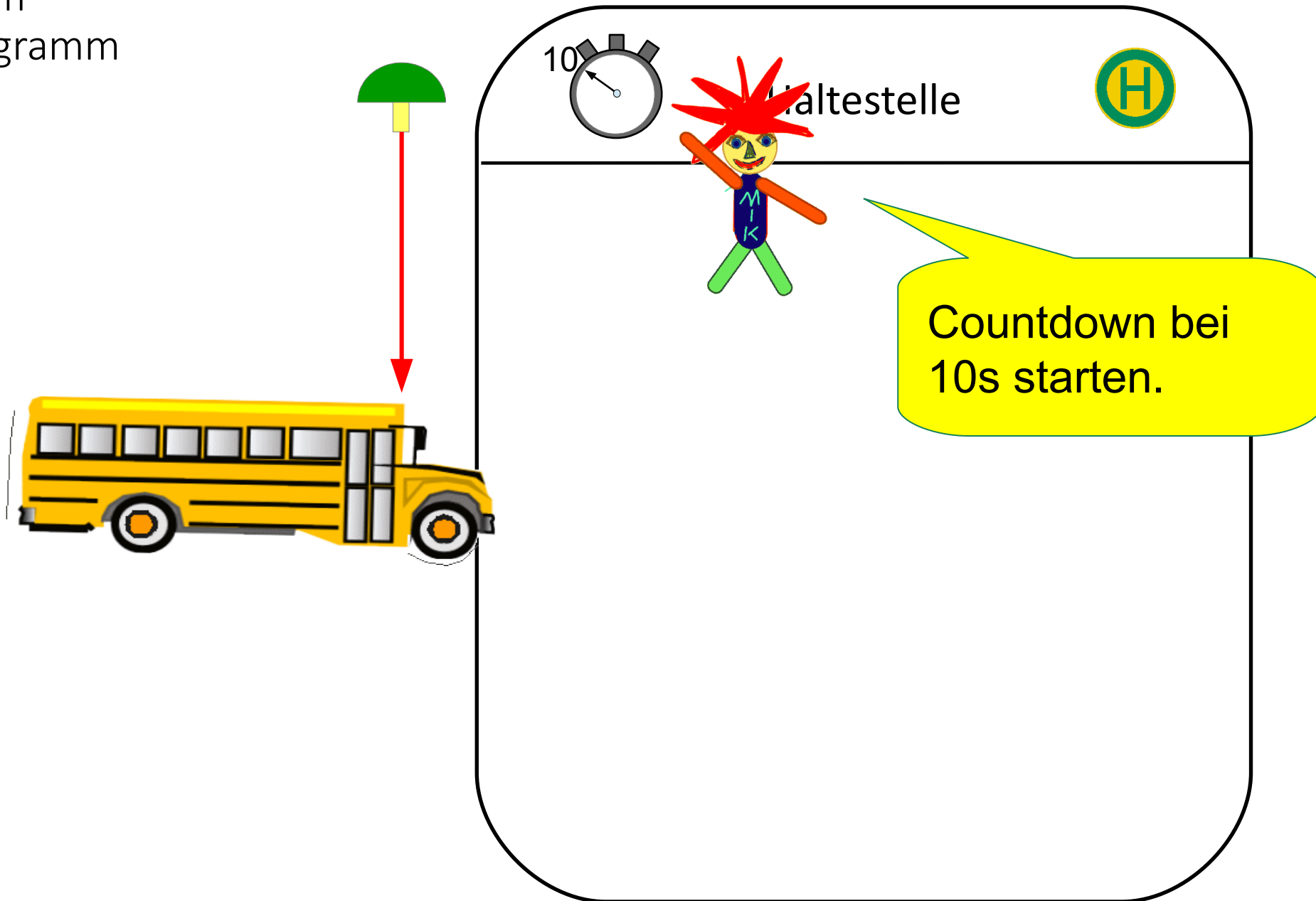
Haltestelle



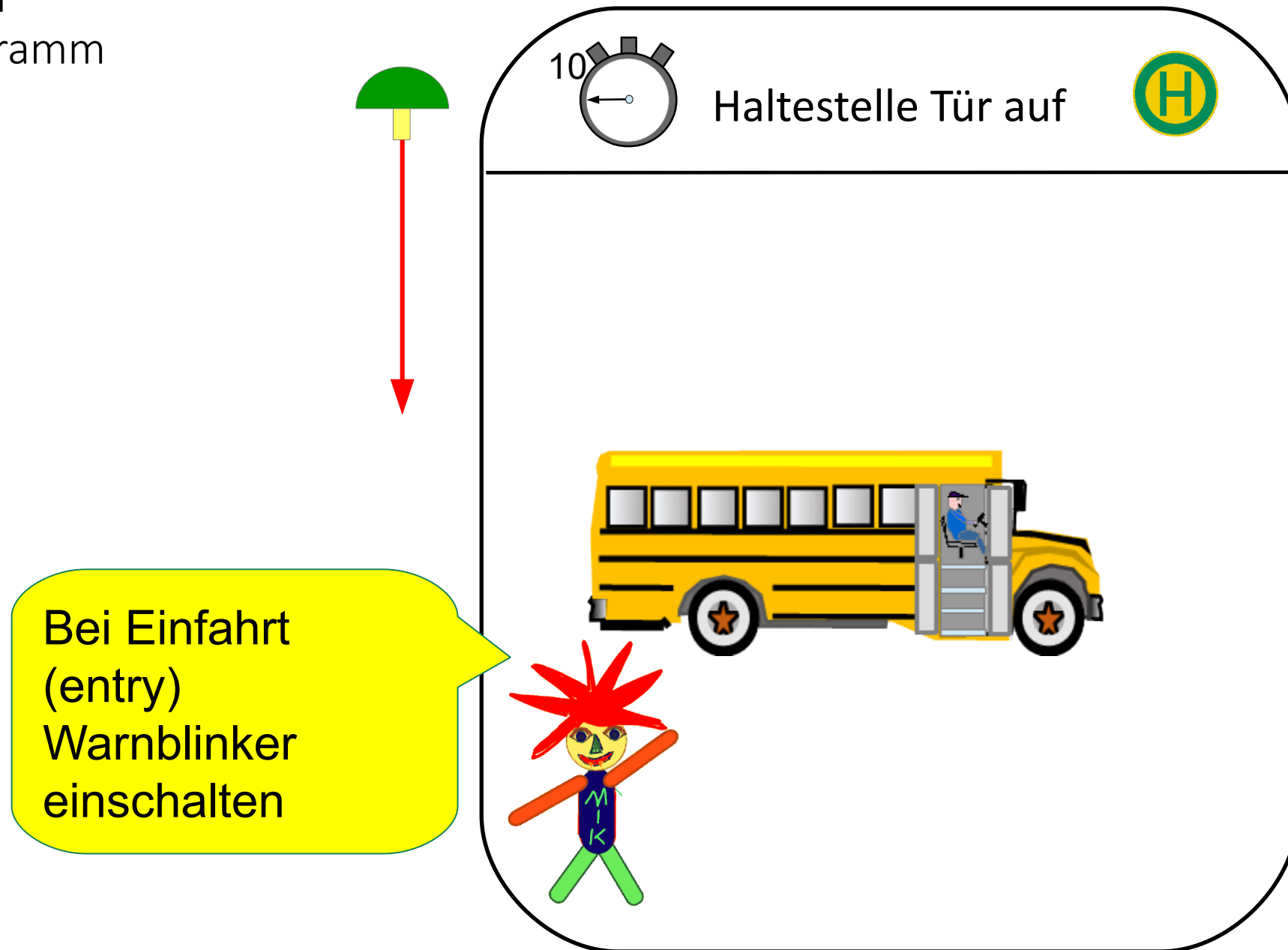
Aktivitäten im Zustandsdiagramm



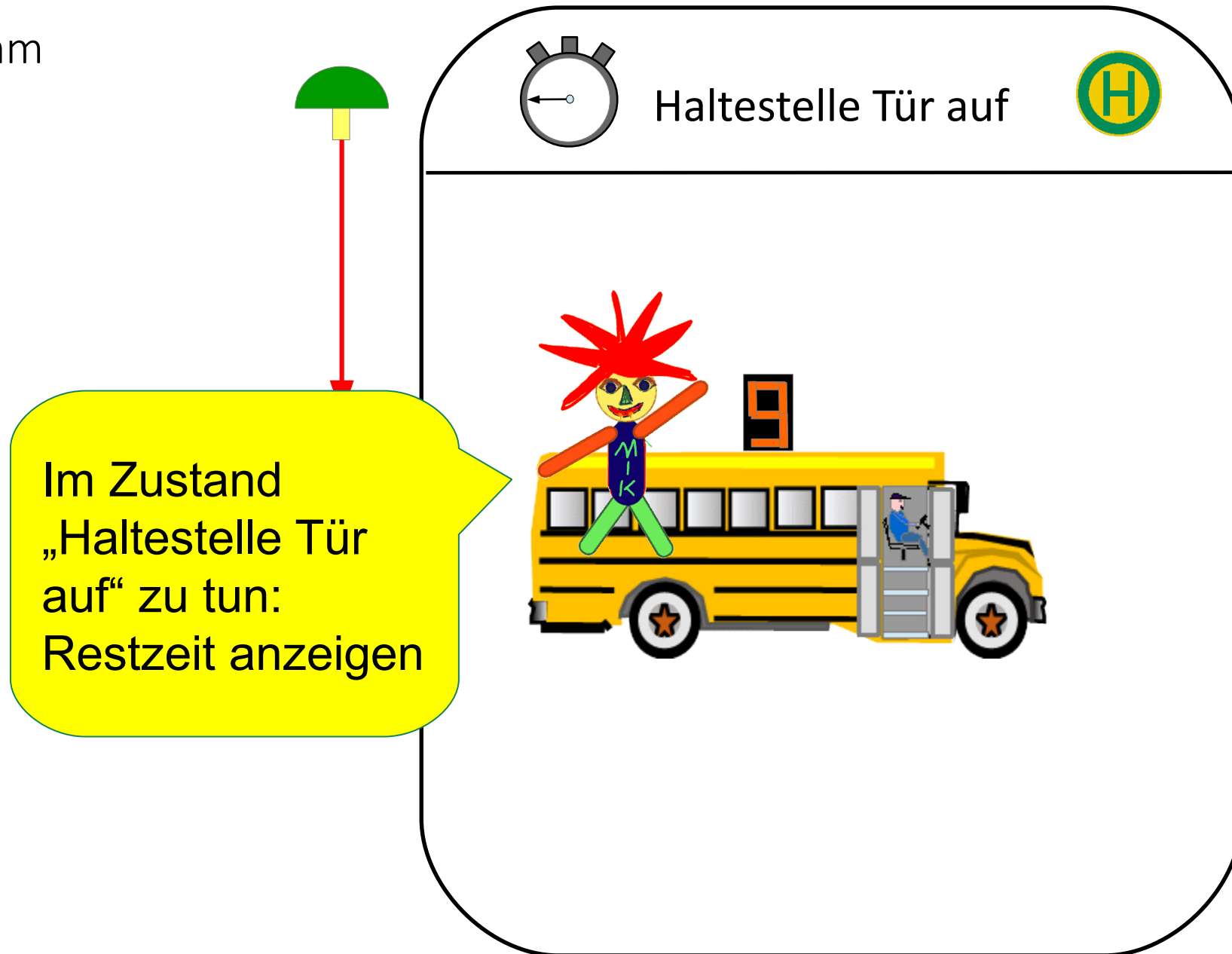
Aktivitäten im Zustandsdiagramm



Aktivitäten im Zustandsdiagramm



Aktivitäten im Zustandsdiagramm

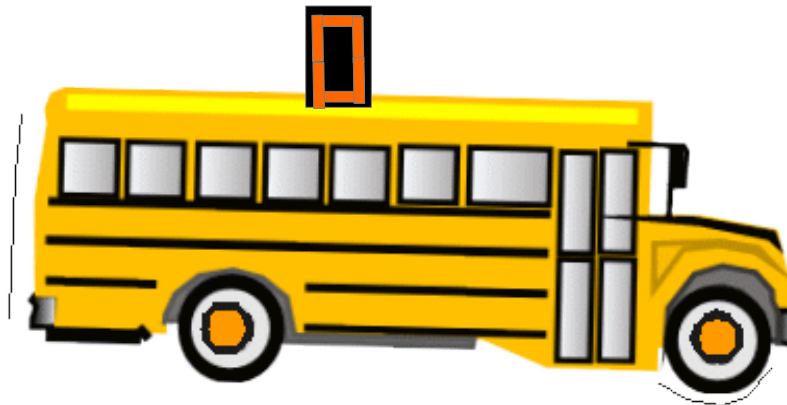


Aktivitäten im Zustandsdiagramm

Bei 0 und Türe
frei:
Tür zu,
Warnblinker aus
losfahren



Haltestelle Tür auf



Aktivitäten im Zustandsdiagramm

Bei 0 und Türe
frei:
Tür zu,
Warnblinker aus
losfahren



Haltestelle Tür auf



Aktivitäten im
Zustandsdiagramm

ankunft()/starteCountDown()



Ereignis: ankunft()
wenn der
Infrarotstrahl
empfangen wird.
Aktivität:
starteCountdown

FahrenTuerZu



HaltestelleTuerAuf



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()



Ereignis: ankunft()
wenn der
Infrarotstrahl
empfangen wird.
Aktivität:
starteCountdown



FahrenTuerZu

HaltestelleTuerAuf



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

FahrenTuerZu

HaltestelleTuerAuf



entry/ warnblinkerEin()



Bei Eintritt (entry) in den Zustand:
„HaltestelleTuerAuf“:
Warnblinker einschalten



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

FahrenTuerZu

HaltestelleTuerAuf



entry/ warnblinkerEin()



Bei Eintritt in den Zustand:
HaltestelleTuerAuf:
Warnblinker
einschalten



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

FahrenTuerZu

HaltestelleTuerAuf



entry/ warnblinkerEin()



Bei Eintritt in den Zustand:
HaltestelleTuerAuf:
Warnblinker
einschalten



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

```
void ankunft()
{
    if (zustand==FahrenTuerZu)
    {
        starteCountDown();
        zustand=HaltestelleTuerAuf;
        warnblinkerEin();
    }
}
```

FahrenTuerZu

HaltestelleTuerAuf



entry/ warnblinkerEin()



Der Code der ISR
ankunft()



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

```
void ankunft()
```

```
{
```

```
  if (zustand==FahrenTuerZu) 1.
```

```
  {
```

```
    starteCountDown();
```

```
    zustand=HaltestelleTuerAuf;
```

```
    warnblinkerEin();
```

```
  }
```

```
}
```

FahrenTuerZu

1.



HaltestelleTuerAuf



entry/ warnblinkerEin()



1. Zustand prüfen.
Das Ereignis darf nur
im Zustand
„FahrenTuerZu“
weitere Auswirkungen
haben



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

```
void ankunft()
```

```
{
```

```
  if (zustand==FahrenTuerZu) 1.
```

```
  {
```

```
    starteCountDown();
```

```
    zustand=HaltestelleTuerAuf;
```

```
    warnblinkerEin();
```

```
  }
```

```
}
```

FahrenTuerZu

1.



HaltestelleTuerAuf



entry/ warnblinkerEin()



Wenn der Zustand „FahrenTuerZu“ ist...



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

```
void ankunft()
```

```
{
```

```
  if (zustand==FahrenTuerZu) 1.
```

```
  {
```

```
    starteCountDown(); 2.
```

```
    zustand=HaltestelleTuerAuf;
```

```
    warnblinkerEin();
```

```
  }
```

```
}
```

FahrenTuerZu 1.



HaltestelleTuerAuf

entry/ warnblinkerEin()



starte den Countdown



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown() 2.

```
void ankunft()
```

```
{
```

```
  if (zustand==FahrenTuerZu) 1.
```

```
  {
```

```
    starteCountDown(); 2.
```

```
    zustand=HaltestelleTuerAuf; 3.
```

```
    warnblinkerEin();
```

```
  }
```

```
}
```

FahrenTuerZu 1.

HaltestelleTuerAuf H

entry/ warnblinkerEin()



wechsle in Zustand „HaltestelleTuerAuf“



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

```
void ankunft()
```

```
{
```

```
  if (zustand==FahrenTuerZu)
```

```
  {
```

```
    starteCountDown();
```

```
    zustand=HaltestelleTuerAuf;
```

```
    warnblinkerEin();
```

```
  }
```

```
}
```

FahrenTuerZu

1.

2.

3.

HaltestelleTuerAuf



entry/ warnblinkerEin()

4.

entry-Aktivität des neuen Zustands:
warnblinkerEin()



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

```
void ankunft()
```

```
{
```

```
  if (zustand==FahrenTuerZu) 1.
```

```
  {
```

```
    starteCountDown(); 2.
```

```
    zustand=HaltestelleTuerAuf; 3.
```

```
    warnblinkerEin(); 4.
```

```
  }
```

```
}
```

FahrenTuerZu

1.

2.

3.

HaltestelleTuerAuf



4.

entry/ warnblinkerEin()



Reihenfolge bei Ereignis:

1.) Ausgangszustand prüfen

2.) Aktivität an der Transition aufrufen

3.) Zustand wechseln

4. Entry-Aktivität des neuen Zustands aufrufen



Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()

FahrenTuerZu

HaltestelleTuerAuf



entry/ warnblinkerEin()
do/ **einaussteigen()**



Im Zustand
„HaltestelleTuerAuf“:
einaussteigen()



Aktivitäten im Zustandsdiagramm

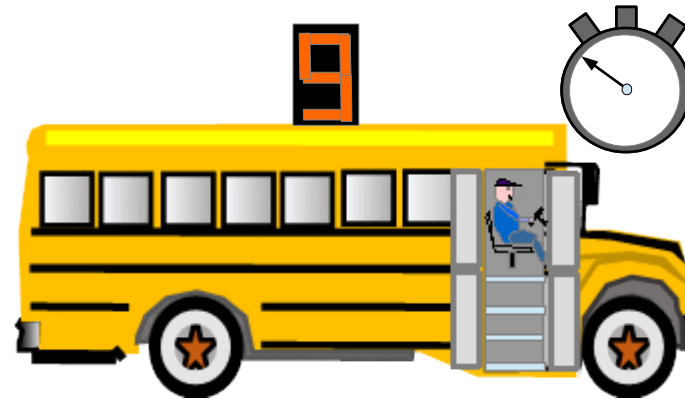
ankunft()/starteCountDown()

FahrenTuerZu

HaltestelleTuerAuf



entry/ warnblinkerEin()
do/ anzeigenRestzeit();

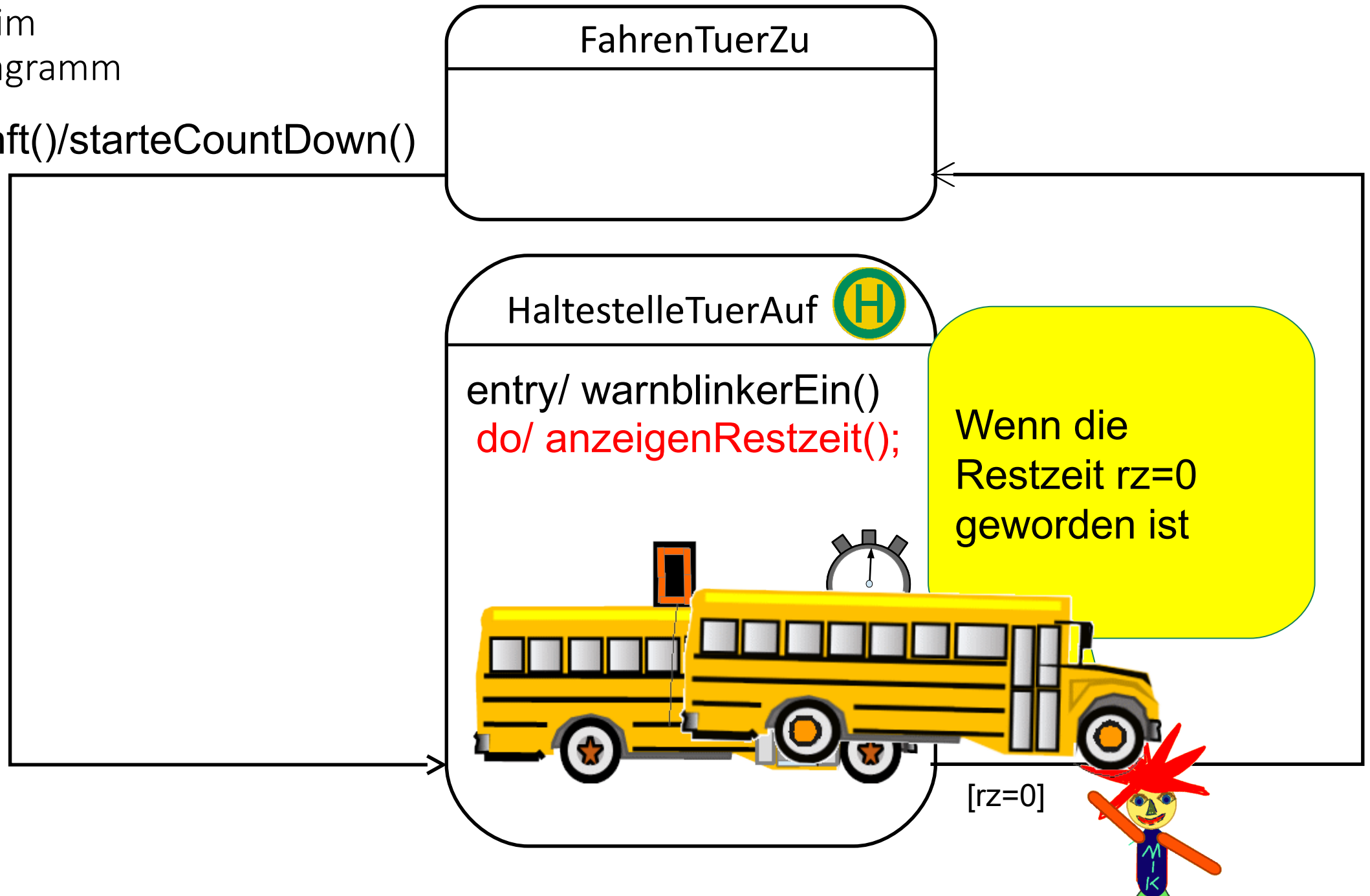


Im Zustand „HaltestelleTuerAuf“:
Die Restzeit anzeigen



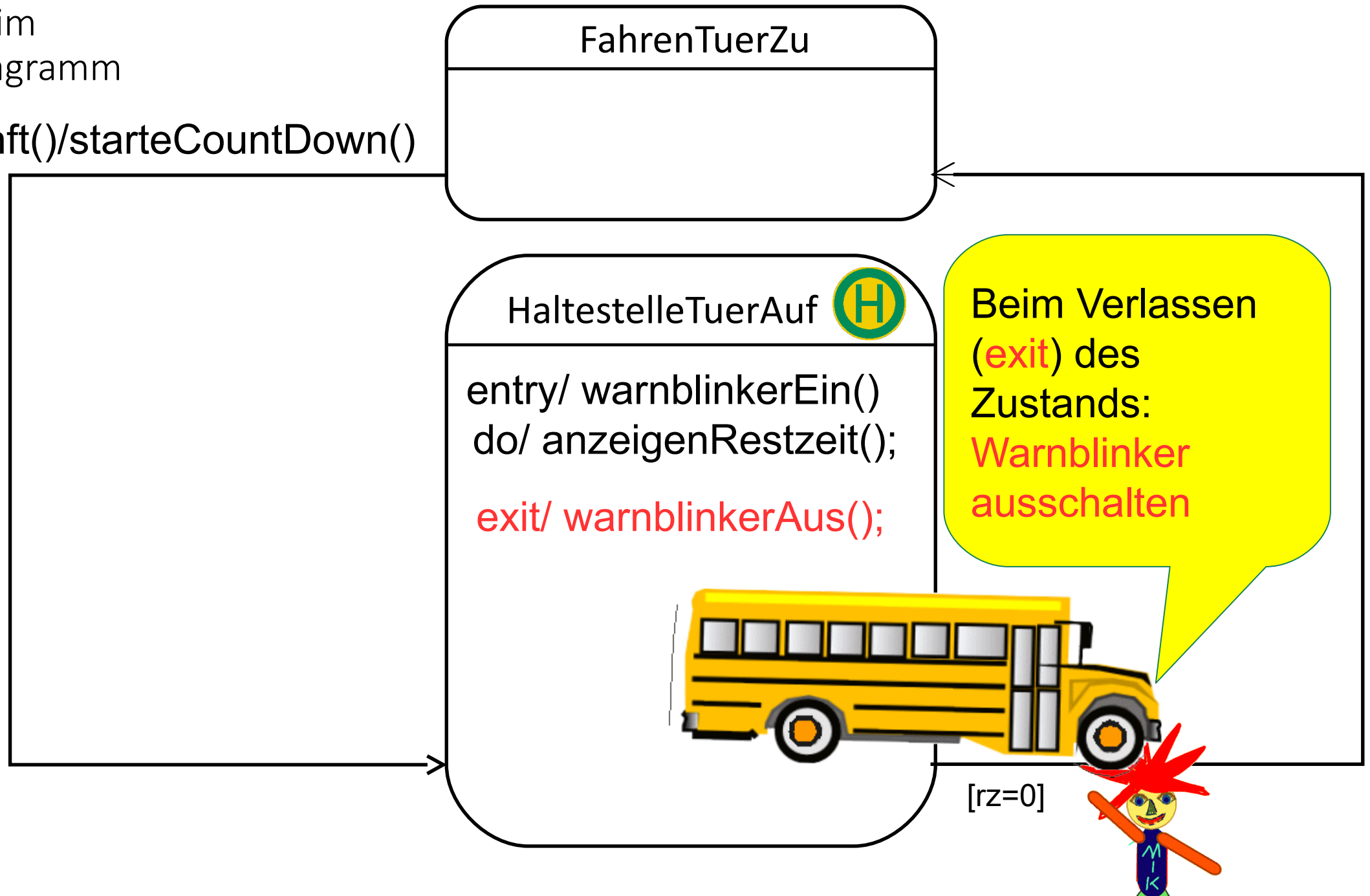
Aktivitäten im Zustandsdiagramm

ankunft()/starteCountDown()



Aktivitäten im Zustandsdiagramm

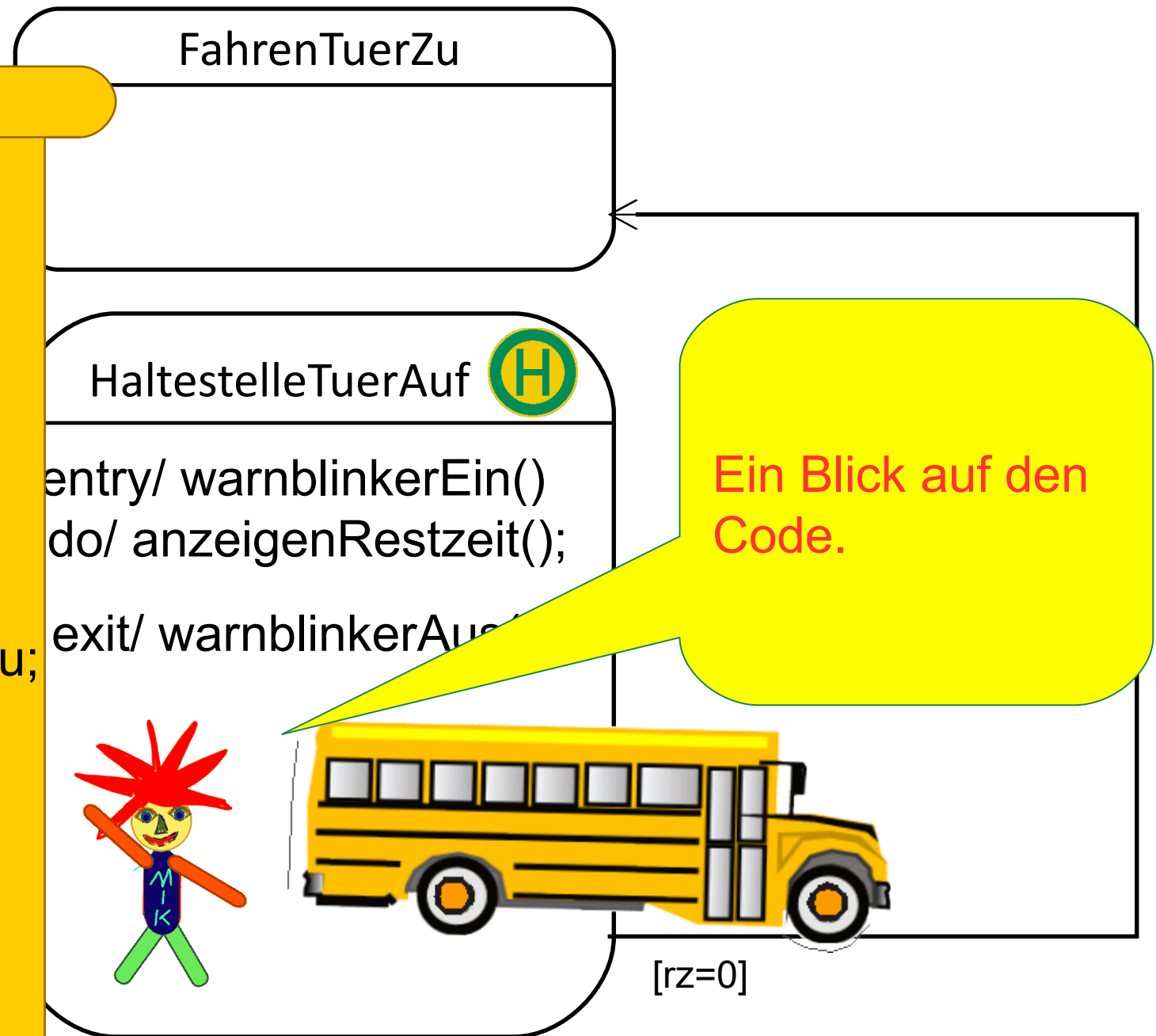
ankunft()/starteCountDown()



Aktivitäten im

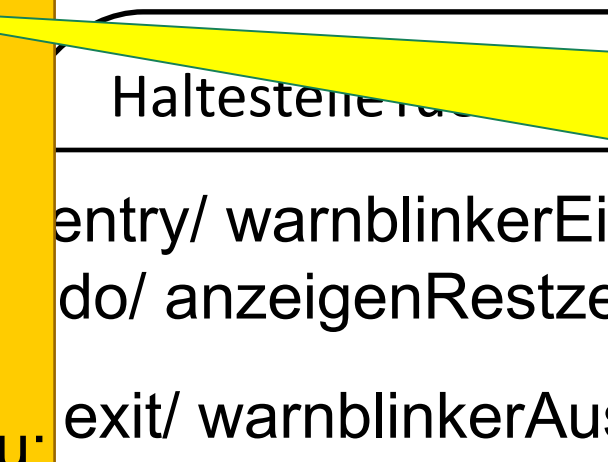
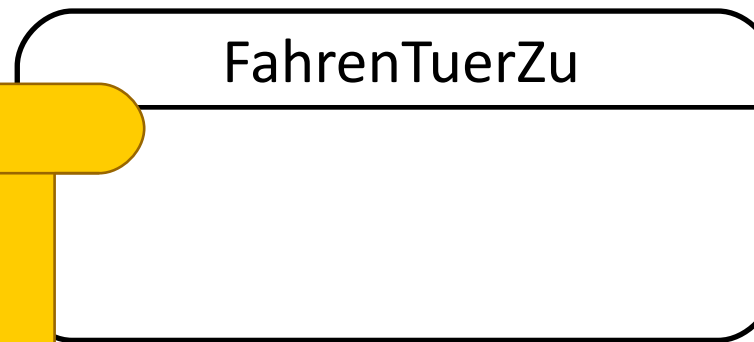
Z

```
int main()
{ init();
  while (true) {
    switch(zustand)
    { case HaltestelleTuerAuf:
      anzeigenRestzeit();
      if(rz==0)
      { warnblinkerAus();
        zustand=FahrenTuerZu;
      }
      break;
    case FahrenTuerZu:
      break;
    }
  }
}
```

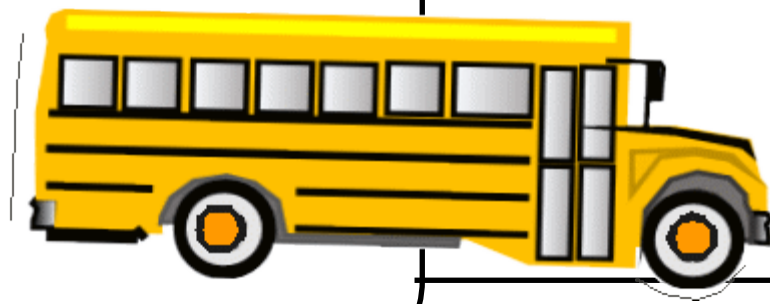


Aktivitäten im
Zustandsautomaten

```
int main()
{ init();
  while (true) {
    switch(zustand) {
      case HaltestelleTuerAuf:
        anzeigenRestzeit();
        if(rz==0)
        { warnblinkerAus();
          zustand=FahrenTuerZu;
        }
        break;
      case FahrenTuerZu:
        break;
    }
  }
}
```



Do-Aktivitäten in der Endlosschleife



[rz=0]

Aktivitäten im

Z

```
int main()
{ init();
  while (true) {
    switch(zustand)
    { case HaltestelleTuerAuf:
      anzeigenRestzeit();
      if(rz==0)
      { warnblinkerAus();
        zustand=FahrenTuerZu;
      }
      break;
    case FahrenTuerZu:
      break;
    }
  }
}
```



1. Zustand prüfen



[rz=0]

Aktivitäten im

Z

```
int main()
{ init();
  while (true) {
    switch(zustand)
    { case HaltestelleTuerAuf:
      anzeigenRestzeit();
      if(rz==0)
      { warnblinkerAus();
        zustand=FahrenTuerZu;
      }
      break;
    case FahrenTuerZu:
      break;
    }
  }
}
```

FahrenTuerZu

HaltestelleTuerAuf



2. Im Zustand
„HaltestelleTuerAuf“



[rz=0]

Aktivitäten im

Z

```
int main()
```

```
{ init();
```

```
  while (true) {
```

```
    switch(zustand)
```

```
    { case HaltestelleTuerAuf:
```

```
      anzeigenRestzeit();
```

```
      if(rz==0)
```

```
      { warnblinkerAus();
```

```
        zustand=FahrenTuerZu; exit/ warnblinkerAus();
```

```
      }
```

```
      break;
```

```
    case FahrenTuerZu:
```

```
      break;
```

```
  }
```

```
}
```

```
}
```

FahrenTuerZu

HaltestelleTuerAuf



3. Do-Aktivität des Zustands „HaltestelleTuerAuf“ ausführen:
anzeigenRestzeit()



[rz=0]

Aktivitäten im

Z

```
int main()
{ init();
  while (true) {
    switch(zustand)
    { case HaltestelleTuerAuf:
      anzeigenRestzeit();
      if(rz==0)
      { warnblinkerAus();
        zustand=FahrenTuerZu;
      }
      break;
    case FahrenTuerZu:
      break;
    }
  }
}
```

FahrenTuerZu

HaltestelleTuerAuf

4. Wächterbedingung
prüfen: [rz=0]



[rz=0]

Aktivitäten im

Zustand

```
int main()
{ init();
  while (true) {
    switch(zustand)
    { case HaltestelleTuerAuf:
      anzeigenRestzeit();
      if(rz==0)
      { warnblinkerAus();
        zustand=FahrenTuerZu;
      }
      break;
    case FahrenTuerZu:
      break;
    }
  }
}
```

FahrenTuerZu

HaltestelleTuerAuf 

entry/ warnblinkerEin()
do/ anzeigenRestzeit();
exit/ warnblinkerAus();



5. Falls erfüllt:
exit-Aktivität des
Zustands aufrufen

Aktivitäten im

Z

```
int main()
{ init();
  while (true) {
    switch(zustand)
    { case HaltestelleTuerAuf:
      anzeigenRestzeit();
      if(rz==0)
      { warnblinkerAus();
        zustand=FahrenTuerZu;
      }
      break;
    case FahrenTuerZu:
      break;
    }
  }
}
```

FahrenTuerZu

HaltestelleTuerAuf



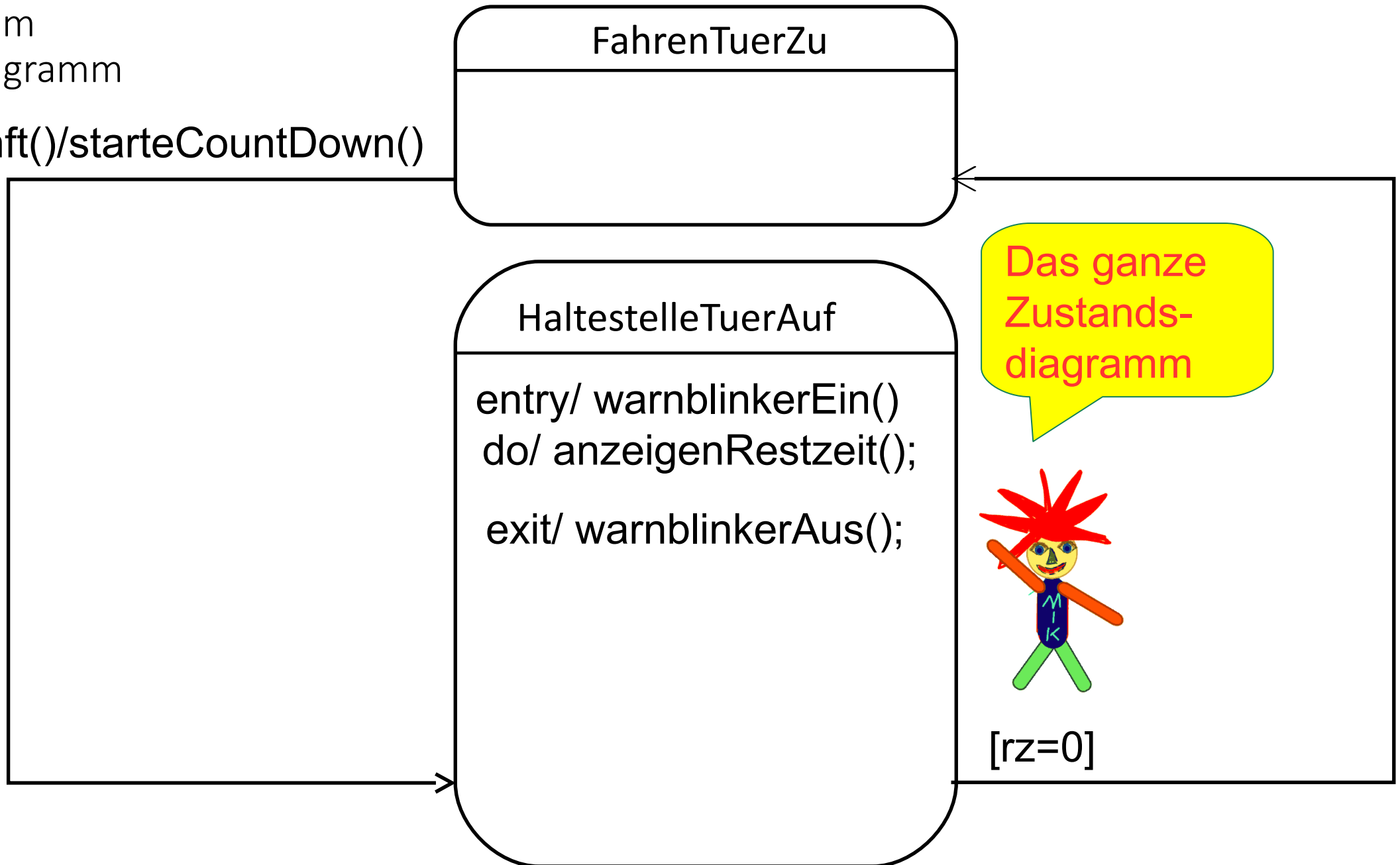
entry/ warnblinkerEin()
do/ anzeigenRestzeit();
exit/ warnblinkerAus();



6. Falls erfüllt:
Dann Zustandswechsel

Aktivitäten im
Zustandsdiagramm

ankunft()/starteCountDown()



Aktivitäten im
Zustandsdiagramm

ankunft()/starteCountDown()

FahrenTuerZu

HaltestelleTuerAuf

entry/ warnblinkerEin()
do/ anzeigenRestzeit();
exit/ warnblinkerAus();

Weiter Aktivitäten:

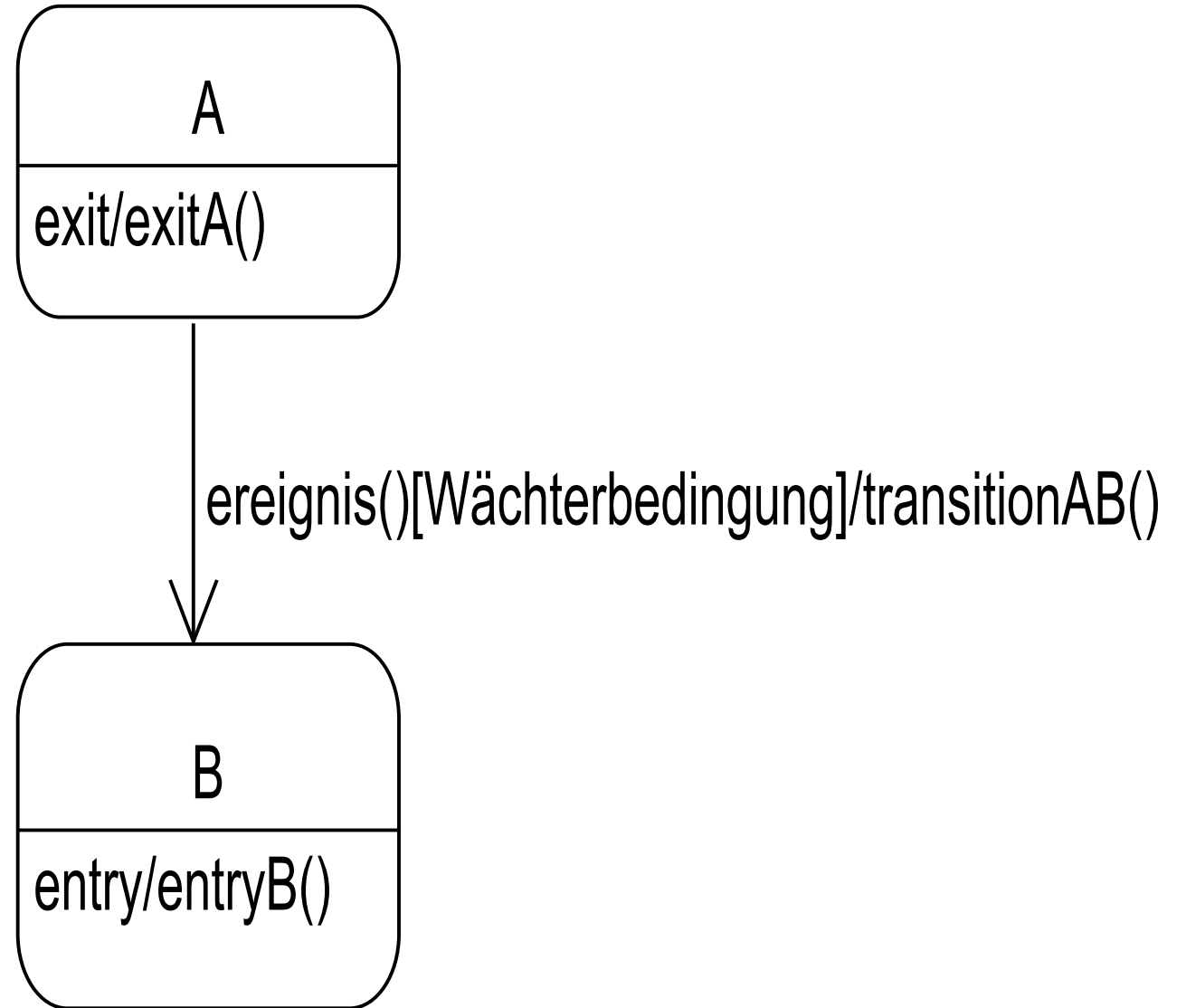
- An der Transition
- Beim Eintritt in den Zustand (entry)
- Beim Verlassen des Zustands (exit)



[rz=0]

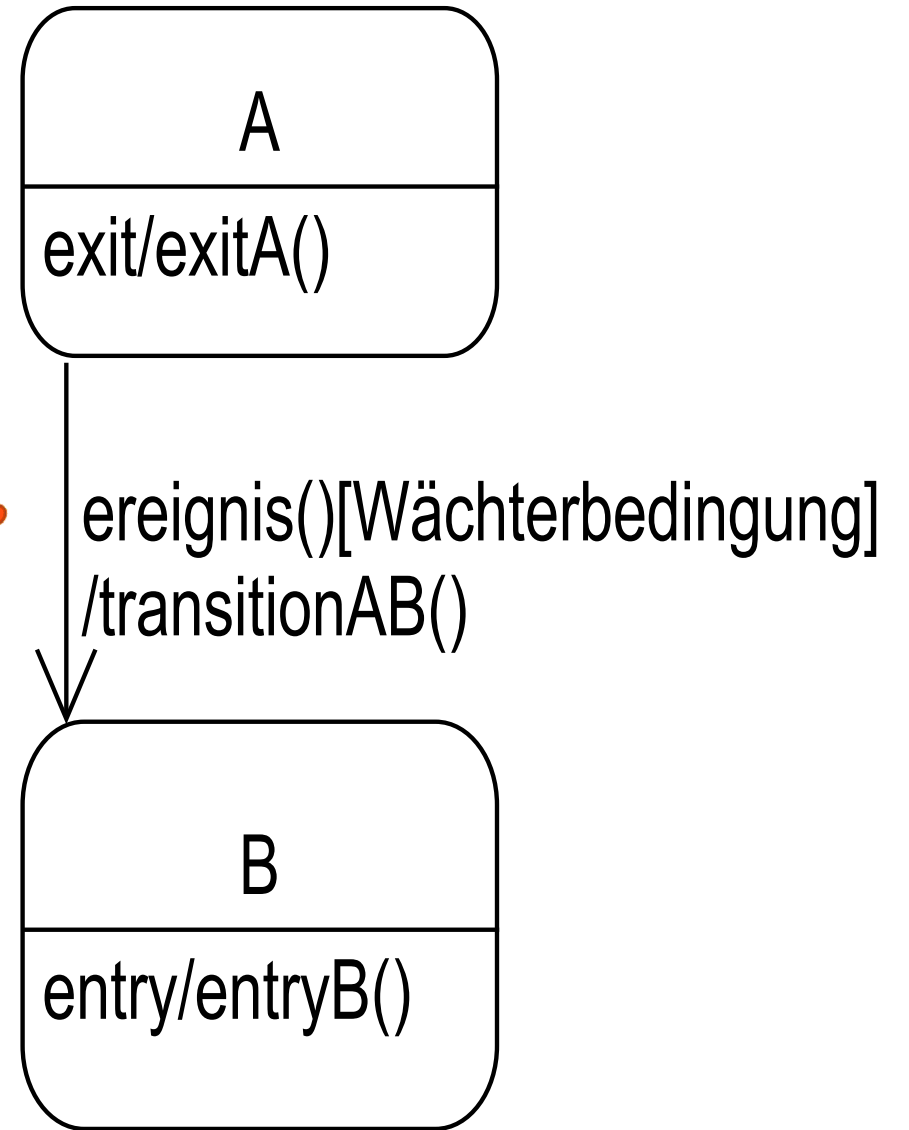
Aktivitäten im Zustandsdiagramm

Allgemein bei Ereignis (Interrupt)



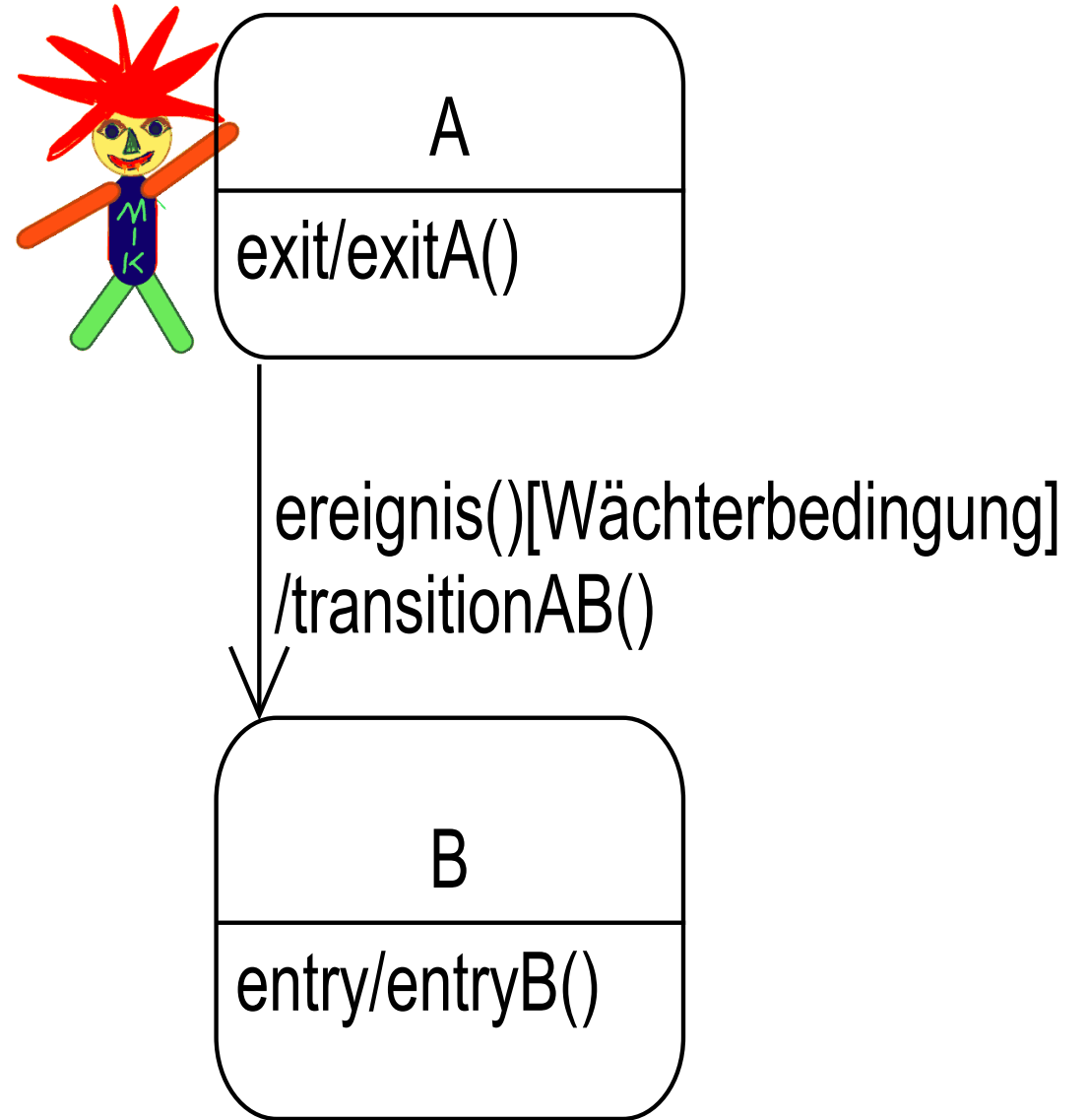
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein



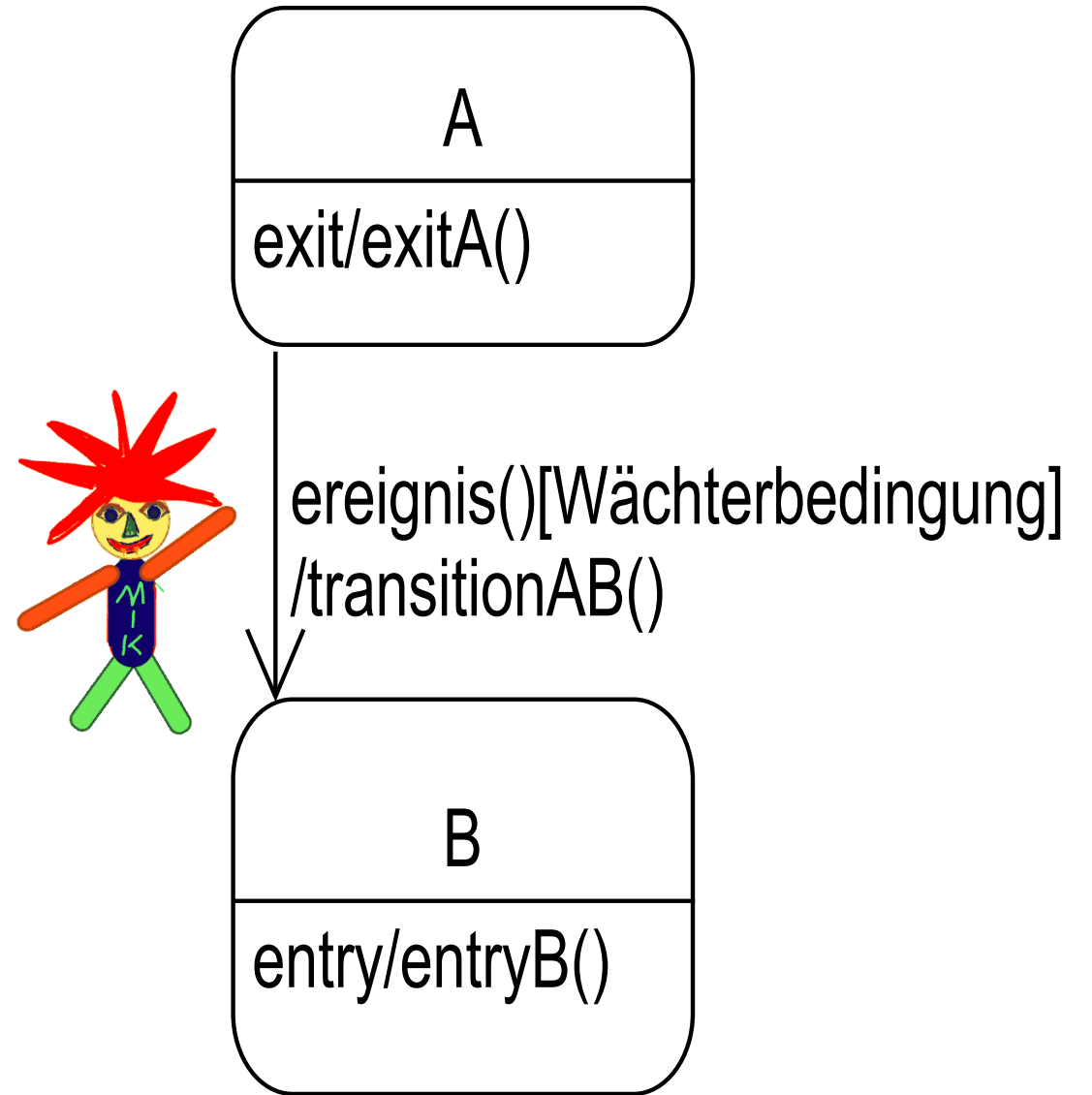
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein
2. Zustand wird geprüft,
nur wenn passt dann:



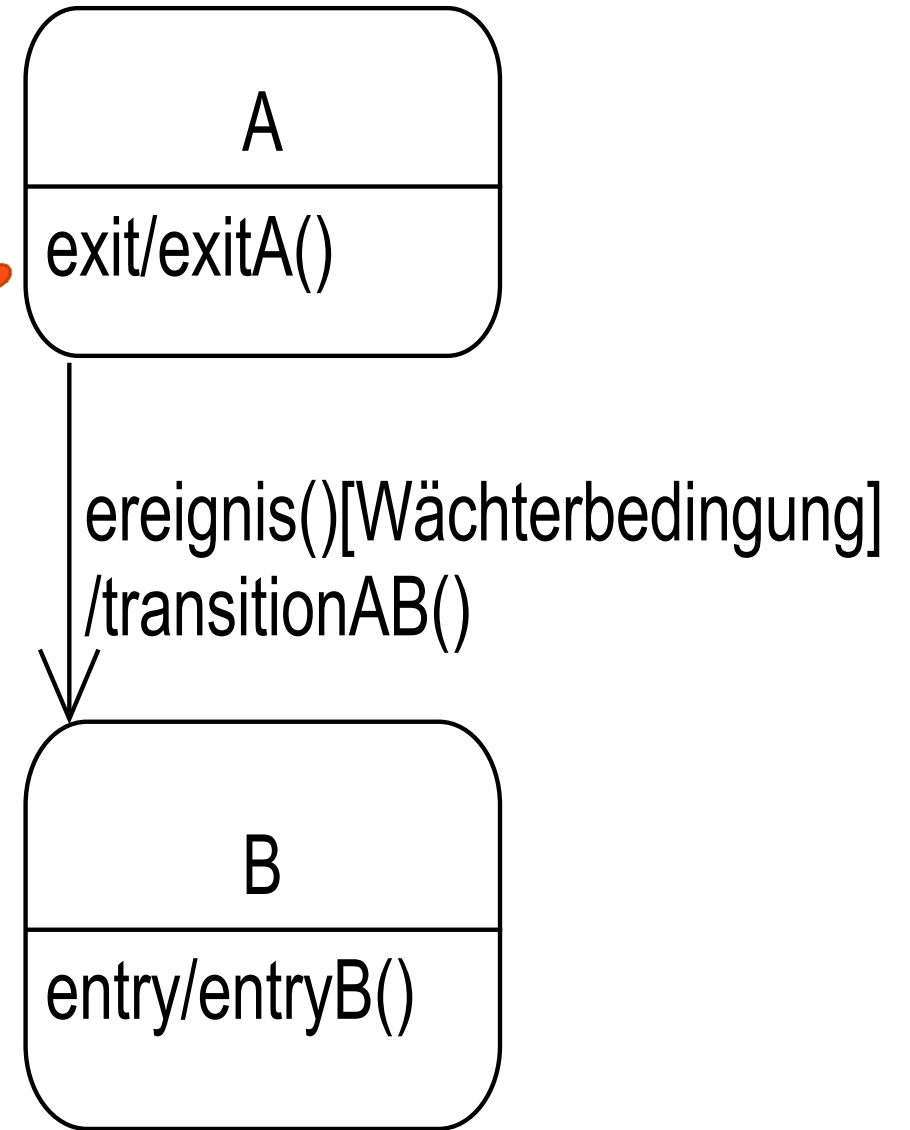
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein
2. Zustand wird geprüft,
nur wenn erfüllt dann:
3. Wächterbedingung prüfen, nur
wenn erfüllt dann



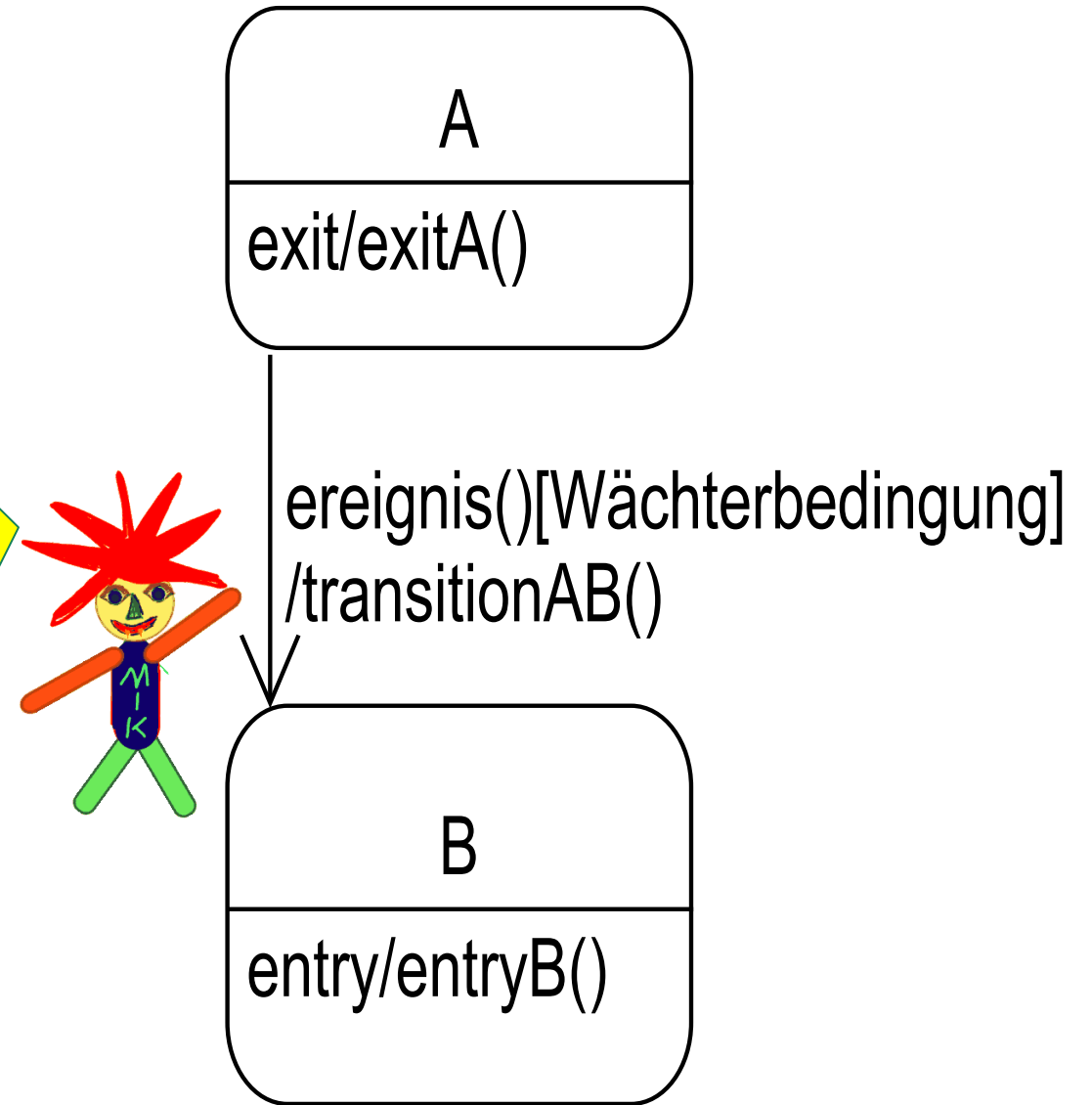
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein
2. Zustand wird geprüft,
nur wenn erfüllt dann:
3. Wächterbedingung prüfen, nur
wenn erfüllt dann
4. exit-Aktivität des Ausgangs-
zustands A exitA() aufrufen



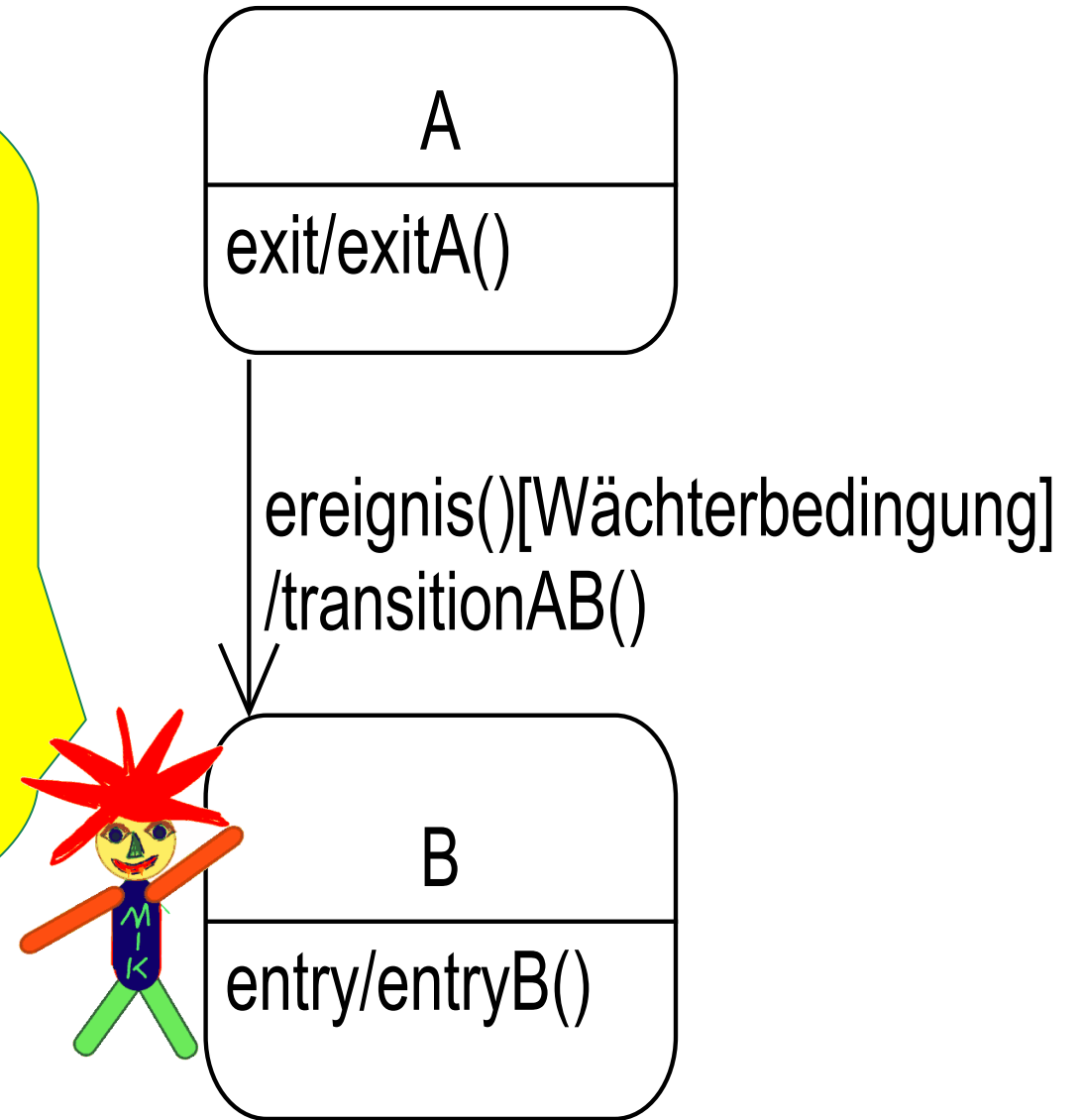
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein
2. Zustand wird geprüft,
nur wenn erfüllt dann:
3. Wächterbedingung prüfen, nur
wenn erfüllt dann
4. exit-Aktivität des Ausgangs-
zustands A exitA() aufrufen
5. Transitionsaktivität ausführen
transitionAB()



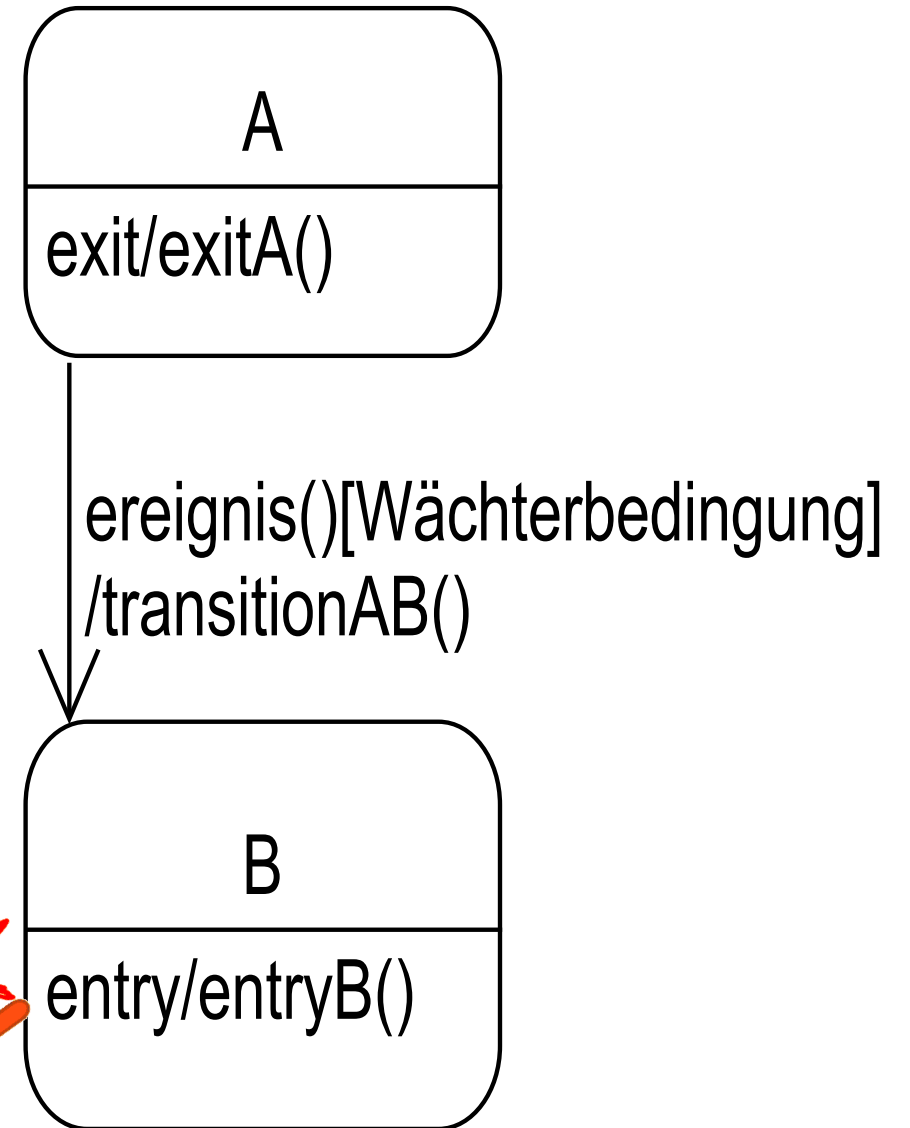
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein
2. Zustand wird geprüft,
nur wenn erfüllt dann:
3. Wächterbedingung prüfen, nur
wenn erfüllt dann
4. exit-Aktivität des Ausgangs-
zustands A exitA() aufrufen
5. Transitionsaktivität ausführen
transitionAB()
6. Zustand wechseln zustand=B



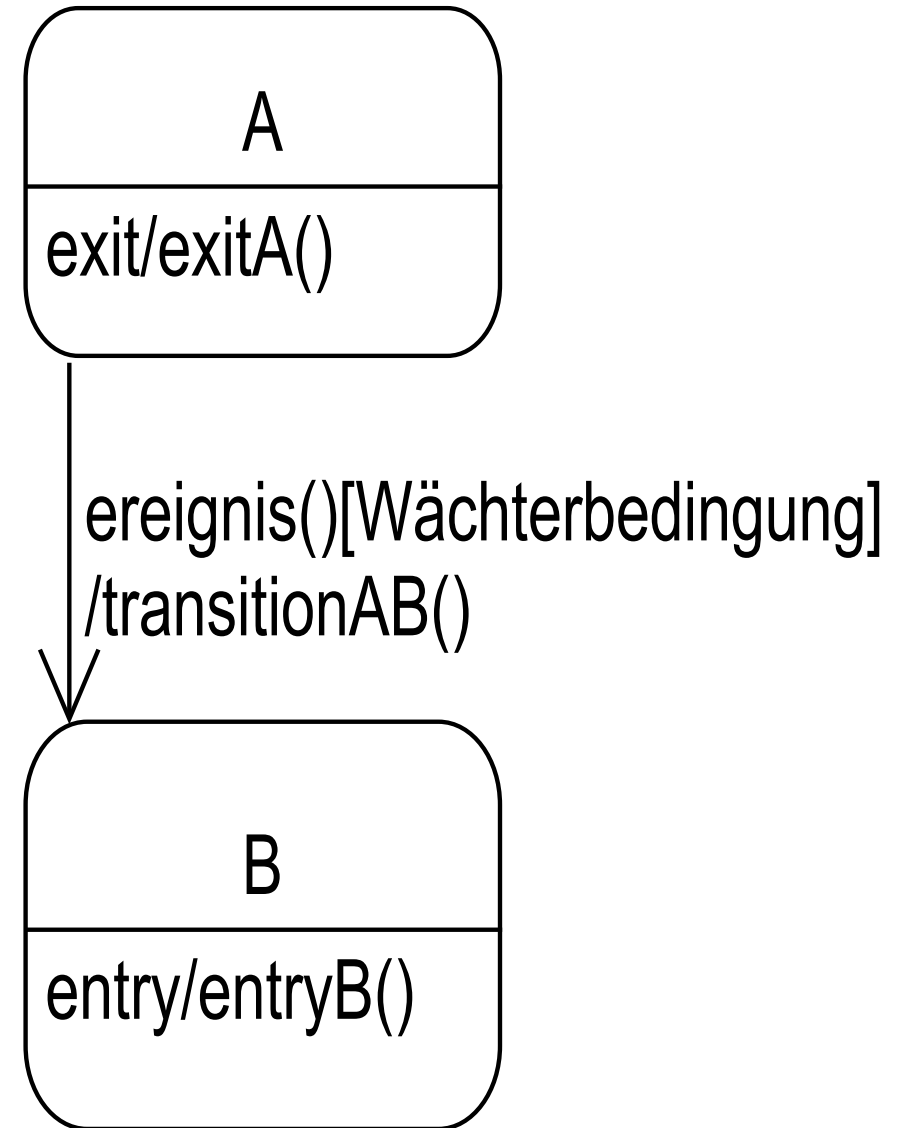
Aktivitäten im Zustandsdiagramm

1. Ein Ereignis ereignis() tritt ein
2. Zustand wird geprüft, nur wenn erfüllt dann:
3. Wächterbedingung prüfen, nur wenn erfüllt dann
4. exit-Aktivität des Ausgangszustands A exitA() aufrufen
5. Transitionsaktivität ausführen transitionAB()
6. Zustand wechseln zustand=B
7. Entry-Aktivität des neuen Zustands B aufrufen: entryB()



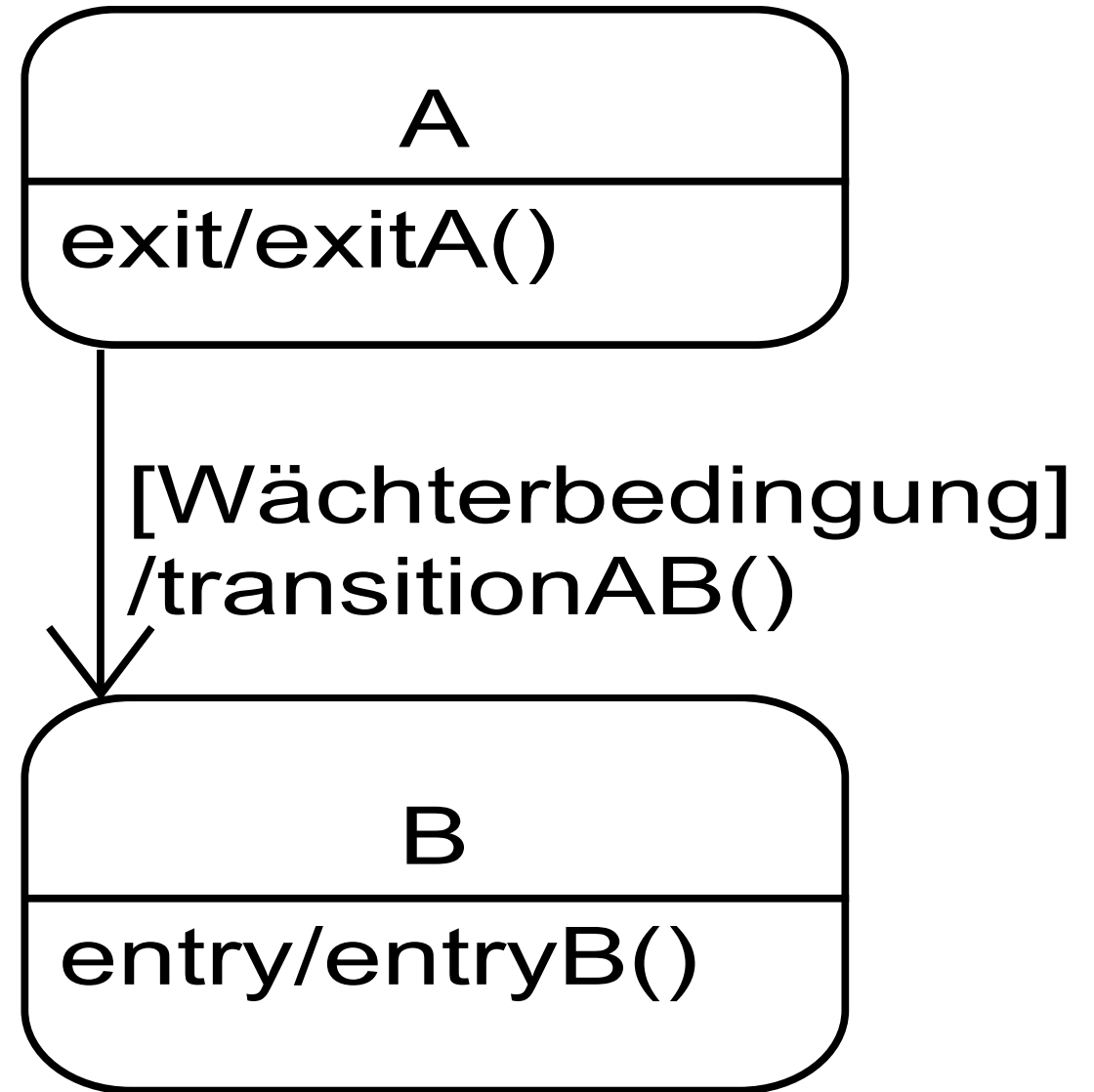
Aktivitäten im Zustandsdiagramm

```
void ereignis()  
{ if (zustand==A) {  
    if (Wächterbedingung) {  
        exitA();  
        transitionAB();  
        zustand=B;  
        entryB();  
    }  
}  
}
```



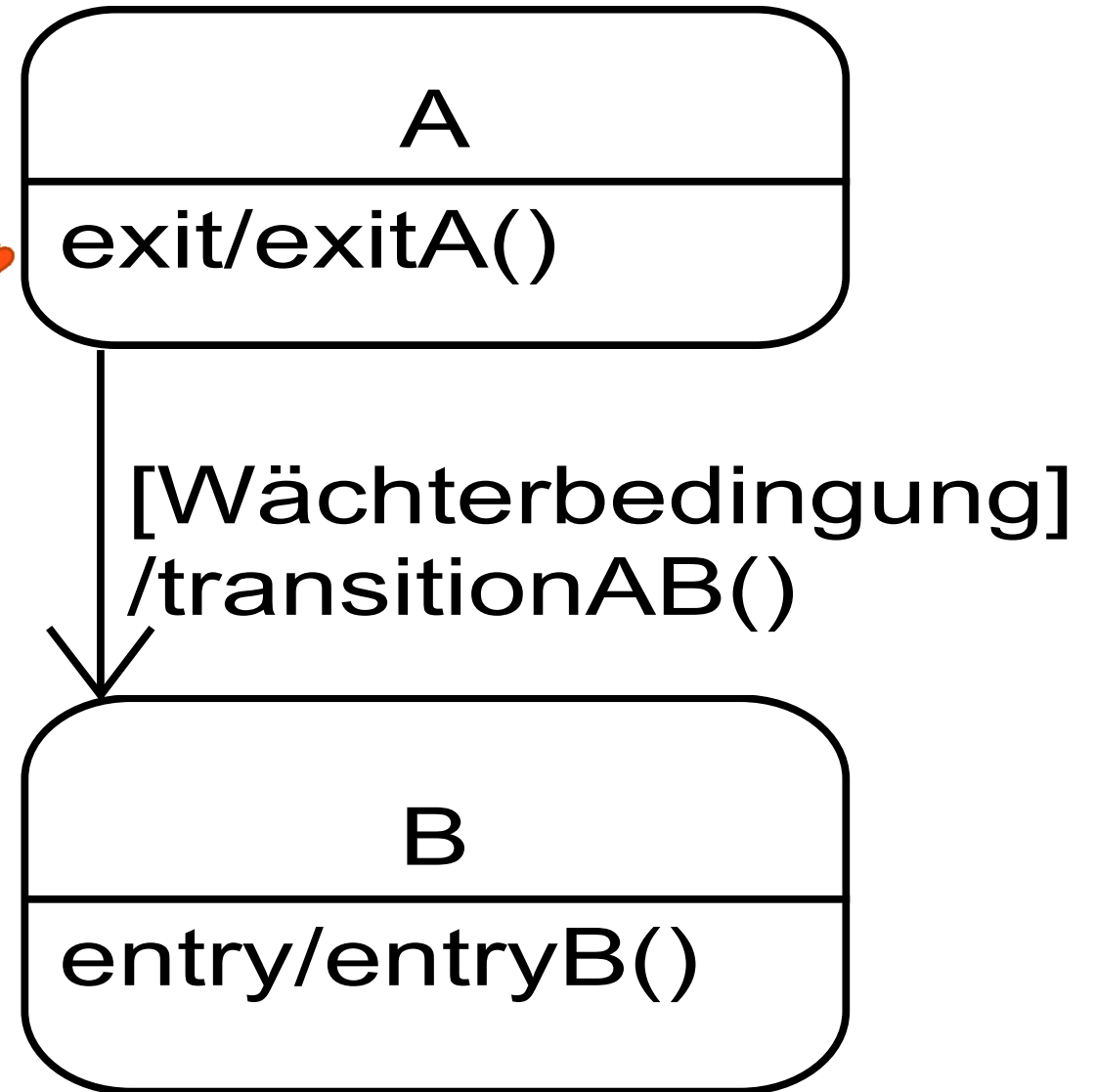
Aktivitäten im
Zustandsdiagramm

Allgemein (ohne
Ereignis)



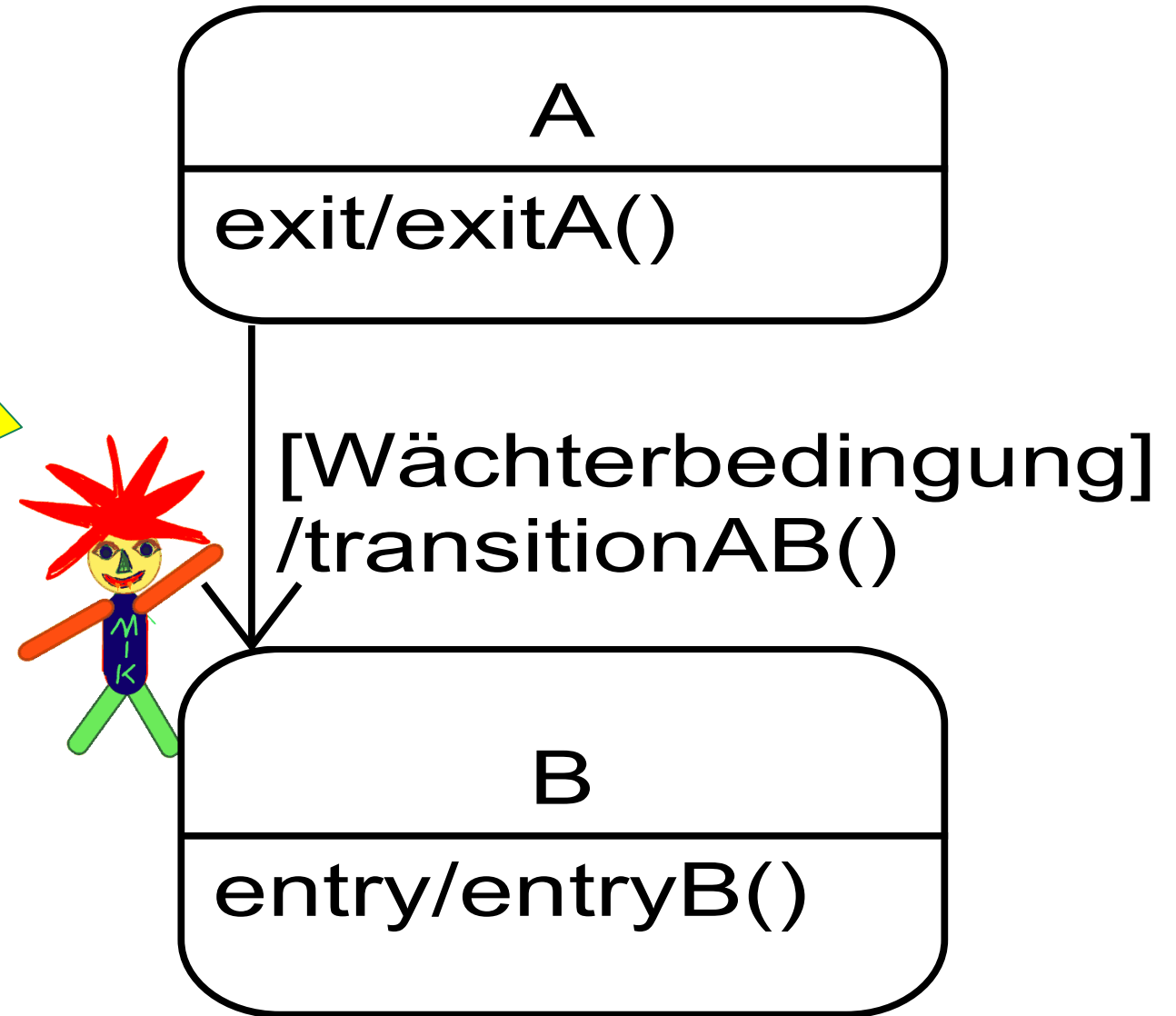
Aktivitäten im Zustandsdiagramm

1. In der Endlosschleife wird im Zustand A die Wächterbedingung geprüft. Nur wenn erfüllt:
2. exit-Aktivität von Zustand A ausführen



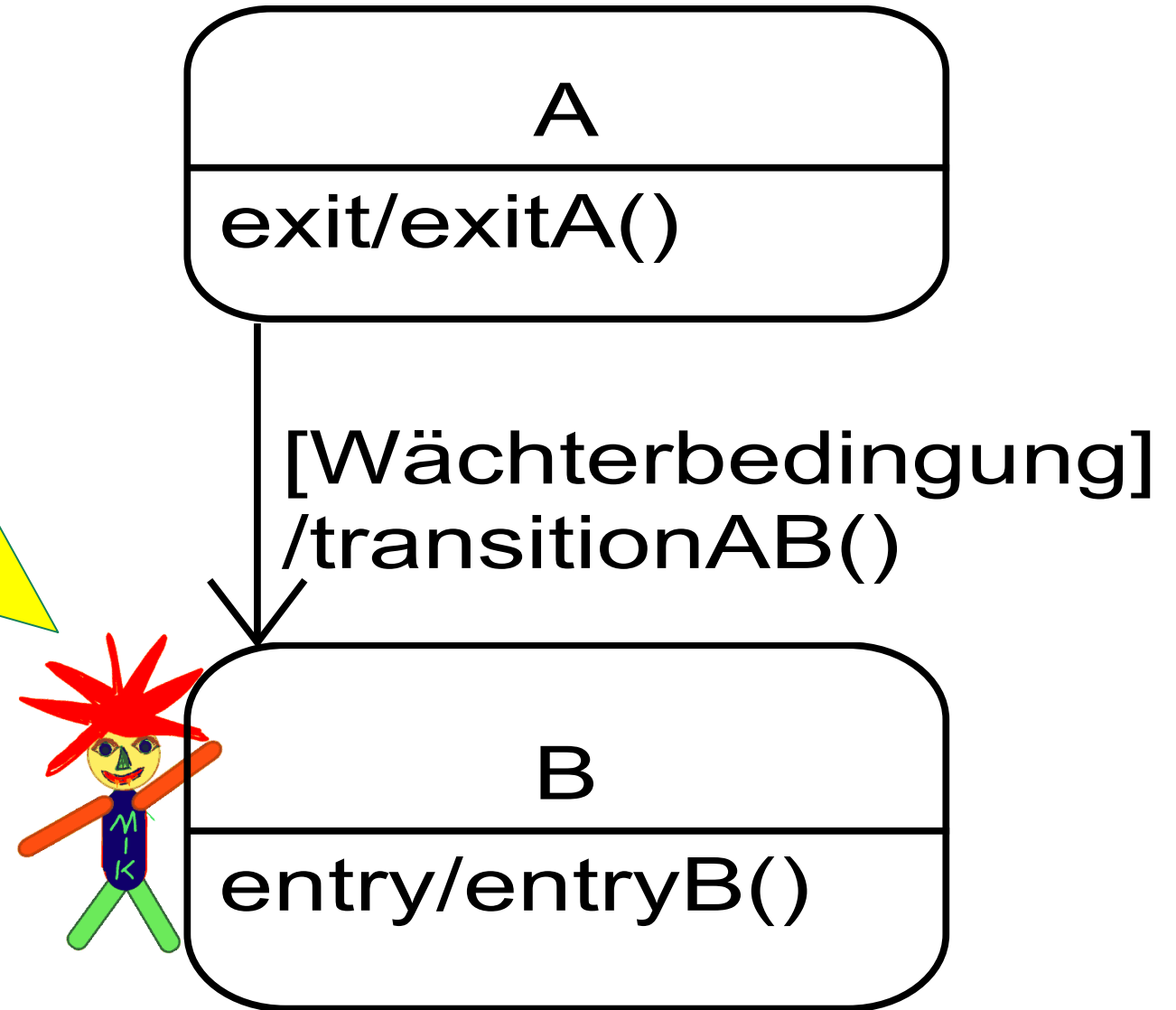
Aktivitäten im Zustandsdiagramm

1. In der Endlosschleife wird im Zustand A die Wächterbedingung geprüft. Nur wenn erfüllt:
2. exit-Aktivität von Zustand A ausführen
3. Transitions-Aktivität ausführen



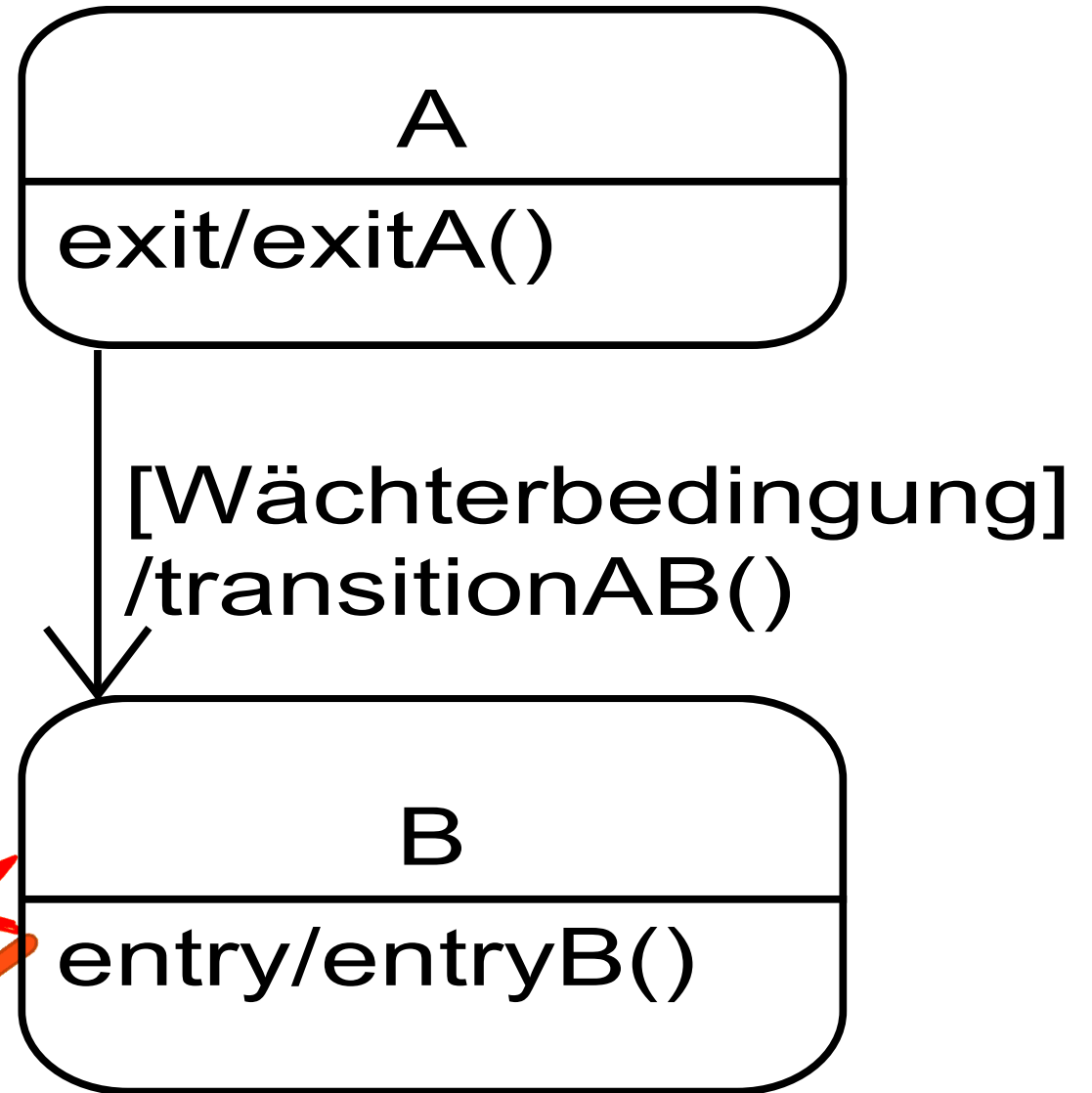
Aktivitäten im Zustandsdiagramm

1. In der Endlosschleife wird im Zustand A die Wächterbedingung geprüft. Nur wenn erfüllt:
2. exit-Aktivität von Zustand A ausführen
3. Transitions-Aktivität ausführen
4. Zustandswechsel: `zustand=B`



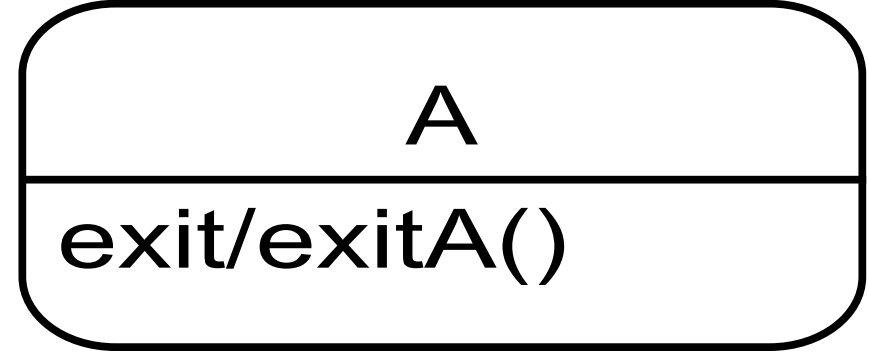
Aktivitäten im Zustandsdiagramm

1. In der Endlosschleife wird im Zustand A die Wächterbedingung geprüft. Nur wenn erfüllt:
2. exit-Aktivität von Zustand A ausführen
3. Transitions-Aktivität ausführen
4. Zustandswechsel: `zustand=B`
5. entry-Aktivität des neuen Zustands B ausführen



Aktivitäten im

```
7 while (true) {  
    switch (zustand) {  
        case A:  
            doA();  
            if (Wächterbedingung) {  
                exitA();  
                transitionAB();  
                zustand = B;  
                entryB();  
            }  
            break;  
        case B: doB(); break;  
    }  
}
```



[Wächterbedingung]
/transitionAB()

