Classic Game Designer (C.G.D) instructions.

Dave Hughes, updated to version 1.3 on 15/10/13

CGD is a utility that runs on the 48k Spectrum and above to create games with that 'old school' early 1980's feel.

Hopefully most of the instructions are within the utility, unless otherwise told key 'E' exits. Apologies in advance for liberal use of the term 'baddy'.

1-Design sprites.

Here you create the moving objects in the game. UDGS 0,1,2,3 are the player sprite right, left, up and down graphics respectively. UDGS 4 – 15 are the 'baddy' sprites, they have only one UDG each so no directional graphics for them.

To edit the player properties (speed, control type) use while editing block 0.

The main keys are;

QAOP-Space – to create your graphic. Keys 1 & 2 to scroll through your sprites.

Key 4 scrolls through the desired effect when the sprite collides with the player. So for this reason you cannot select it for UDGs 0-3 (the player sprite). There are 4 types of effects:

1. Do nothing -does nowt
2. Kill player – player loses a life on collision
3. Kill guardian – the baddy sprite is switched off & counter 2 is decreased.
4. Dec counter 1 – counter 1 is decreased.

Key 5 – scrolls through the sprite speed, 9 is the fastest, 1 is very, very slow.

Key 6 – scrolls through the control mechanism. For the player it is the type of key read, for the baddies it is the AI. It is best to change the player conditions on UDG 0.

Key 'G' generates a random mirrored UDG, it may on occasion look like a space invader. It overwrites the current UDG.

The others keys are hopefully self-explanatory, not mentioned is key 'R' which can be used to rotate the sprite.

2- Design blocks

You have 32 (0-31) blocks to make you game. QAOP-Spc to make you graphics, keys 1 & 2 to scroll through the blocks.

Block 31 is enlarged and used on the game intro screen. It can also be used as an in game block.

Keys 5 & 6 scroll through the effect the block will have on the player. Keys T & Y scroll through the effect the block with have on the baddy.

There are quite a few **block effects**:

- **Do nothing**: passive block you can move through with no effect
- **Solid block**: sprite cannot move through block
- **Change baddy types**: changes baddy AI on collision with this block type. Does not affect player even when selected.
- **Make sprite down**. Affects baddys only, makes a unidirection sprite. Same for up/rt/left
- **Make sprite up.**
- **Make sprite right.**
- **Make sprite left.**
- **Make sprite random.** Baddy only, makes unidirectional in a random direction
- **Dec counter 1**: counter 1 is a 16 bit counter (0-65535). Collision with this block decreases it by 1 and deletes the block with block # 1.
- **Inc counter 1**: counter 1 is a 16 bit counter. Collision with this block increases it by 1 and deletes the block with block # 1.
- **Dec counter 2**: counter 2 is a 16 bit counter. Collision with this block decreases it by 1 and deletes the block with block # 2.
- **Inc counter 2**: counter 2 is a 16 bit counter. Collision with this block increases it by 1 and deletes the block with block # 2
- **Dec counter 3**: counter 3 is an 8 bit counter (0-255). Collision with this block decreases it by 1 and deletes the block with block # 3.
- **Inc counter 3**: counter 3 is an 8 bit counter. Collision with this block increases it by 1 and deletes the block with block # 3
- **Zero counter 1**: zeroes counter and deletes with block # 1
- **Zero counter 2**: zeroes counter and deletes with block # 1
- **Zero counter 3**: zeroes counter and deletes with block # 1
- **Force down**: forces player down one square, unable to return up
- **Force up**: forces player up one square, unable to return down
- **Force left**: forces player left one square, unable to return right
- **Force right**: forces player right one square, unable to return left
- **Counter 1 =0, locked**: if counter 1 is zero it cannot be moved through
- **Counter 2 =0, locked**: if counter 1 is zero it cannot be moved through
- **Counter 3 =0, locked**: if counter 1 is zero it cannot be moved through
- **Counter1<>0, locked**: if counter 1 is not zero it cannot be moved through
- **Counter2<>0, locked**: if counter 1 is not zero it cannot be moved through
- **Counter3<>0, locked**: if counter 1 is not zero it cannot be moved through
- **Counters123nzlocked**: counters 1,2 & 3 must be zero to allow passage
- **Setflag1 ON**: sets flag 1 ON, deletes with block 1
- **Setflag2 ON**: sets flag 2 ON, deletes with block 2
- **Setflag3 ON**: sets flag 3 ON, deletes with block 3
- **Setflag1 OFF**: sets flag1 OFF, deletes with block 1
- **Setflag2 OFF**: sets flag2 OFF, deletes with block 2
- **Setflag3 OFF**: sets flag3 OFF, deletes with block 3
- **Set all flags ON**: sets flags 1,2,3 ON, deletes with block 0

- **Set all flags OFF**: sets flags 1,2,3 OFF, deletes with block 0
- **Toggle ALL flags**: if a flag is ON it is turned OFF, and vice versa, for all 3 flags. Deletes with block 0
- **Flag 1 ON = locked**: if flag 1 is ON this block cannot be moved through
- **Flag 2 ON = locked**: if flag 2 is ON this block cannot be moved through
- **Flag 3 ON = locked**: if flag 3 is ON this block cannot be moved through
- **Flag 1 OFF = locked**: if flag 1 is OFF this block cannot be moved through
- **Flag 2 OFF = locked**: if flag 2 is OFF this block cannot be moved through
- **Flag 3 OFF = locked**: if flag 3 is OFF this block cannot be moved through
- **Open if timer = 0**: only if the timer is zero this block can be moved through
- **Open if timer<>0**: if timer is zero this block cannot be moved through
- **Decrease timer**: decreases timer by 1, block is not deleted
- **Increase timer**: increases timer by 1, block is not deleted
- **Next level up**: completes the current level and jumps to the next one.
- **Next level down**: takes the player down one level.
- **Goto level 1**: these levels can also be used as e.g. teleporters or screen edges to give you any map shape you want in a multiscreen game.
- **Goto level2**
- **Goto level 3**
- **Goto level 4**
- **Goto level 5**
- **Goto level 6**
- **Gotot level 7**
- **Goto level 8**
- **Goto level 9**
- **Complete game**: whichever level the player is on, this successfully finishes the whole game.
- **Kill sprite**: turns the sprite off, this does not work well if applied to the player but works better when applied to the baddies. Counter 2 is decreased.
- **Kill all baddies**: turns off all baddies on the current screen. They are active on return to the screen.
- **Kill player**: player loses a life
- **Extra life**: increases lives by one, makes zap noise but does not interrupt game
- **Gravity ON**: set gravity on, pulls the player down slowly, the speed can be changed (see POKES). Rem if gravity is mixed with 'force' blocks you can get ghost graphics.
- **Gravity OFF**: turns gravity off, whatever the current condition.
- **Toggle gravity**: if gravity currently on it turns off, and vice versa
- **Pacman tablet**: gives player immunity, sets the baddies to run away mode, temporary effect, border flash and tick noise
- **Immunity**: gives player temporary immunity, border flash and tick noise
- **Slow block**: sprite movement is slowed while passing through
- **Next block down**: when sprite passes over it changes to the next block number down
- **Next block up**: when sprite passes over it changes to the next block number up
- **Push block**: can be pushed through block 0 only. Push blocks automatically screen wrap so if you do not want this you will need to create your own solid block border.
- **Push 'n' squash**: can be pushed freely through block 0, but squashes & disappears from the map on contact with any other. Push blocks automatically screen wrap so if you do not want this you will need to create your own solid block border.

- **Push ‘n’ lock**: a specialist block, a bit like a Sokoban block. The push block can only be pushed through block 0, and when it comes into contact with block 4 it becomes block5 and counter 3 is decreased. For this reason if using be sure to keep blocks 4 & 5 free. Push blocks automatically screen wrap so if you do not want this you will need to create your own solid block border.
- **Print message 1**: prints a 32 character message on the screen. This message is set in option 8 ‘Edit game text’. Same for message 2 & 3.
- **Print message 2**
- **Print message 3**

3- Font designer

A not very sophisticated designer*. QAOP-Space to make the chars, 1 & 2 to scroll through them. 'B' makes the entire font bold, 'I' makes the entire font italic. ‘R’ removes your work and replaces it with the regular Sinclair font.

So be careful, it is very easy to spoil your font with one key press.

If you want to load you own font the 768 bytes should be inserted at memory address 52451

**considerably better font editors are available.*

4- Map designer

Follow the instructions to edit the 9 screens available.

Not all the keys are listed in the utility:

Keys QAOP-Space to move around the map and insert blocks.

Keys 1 & 2 scroll through the block you want to insert. It moves through the entire 0-32 blocks plus the sprites. So to insert the player or a baddy you need to find them in these blocks and insert it in position.

Key X – clears the entire screen with block 0.

Key C – copies the entire screen to the clipboard

Key V – pastes the contents of the clipboard into the current screen edit

So be careful, keys X and V can ruin your work with one key press

5- Event manager

Sets some conditions for game events. It is possible to make a playable game using blocks only, so it is safe to leave them all as ‘do nothing’ if you want.

Key 1: scrolls through the effect that leads to the level being successfully completed. E.g. a frogger style game can be created with the ‘player at top’.

Key 2: scrolls through the effect that leads to the level failing and the player losing a life.

Keys 3 & 4: scrolls through the effect when the player or baddy meet the screen edge.

- Screen wrap: on meeting the screen edge the sprite appears at the opposite edge
- Random wrap: on meeting the screen edge the sprite appears at a random position at the opposite edge
- Next screen: makes the game a flip screen game, the next screen along or vertically is moved to.

6- Item counters

Sets the game counters 1,2,3 to a given number at the start of a level. Useful for collectable objects that manipulate the counters.

N – Set the counters to a fixed number at the start of each level

Use keys 1 & 2, Q & W and A & S to inc and dec counters 1,2,3 respectively.

B – sets the counters to a particular block abundance at the start of each level.

This function saves you having to count a particular block that is large in number.

At the start of a level the program counts the numbers of the selected block and sets the counter to the same number. This saves laborious counting, and allows a different number per level (unlike the fixed number choice).

For this reason you can only set all counters to a fixed number, or blocks, not a combination of the two.

7 – Variables

Use the keys specified to inc and dec the variables on the screen.

If timer is set to zero it is not printed on the screen during a game.

If doing a flip-screen game it is best to set level to 1.

8-Game text

This is the text that appears at the bottom of each level/screen.

Press the level number you want to edit, or B, G, C for bottom line, (game over message, congrats screen), then type the text. Press A, S, D to edit in game message 1,2,3 respectively. While typing pressing ‘ENTER’ returns.

9-Effects

You have two effects to include when the game is completed or the game is over. Press the keys specified to scroll through the (limited) number of effects, the on screen descriptions should be self-explanatory.

I – Introscreen text

This is the paragraph that appears at the bottom of the game intro screen. You have 256 character spaces for this. Press 'ENTER' to exit.

C – Game colours

Follow the on screen instructions to set the game paper and ink colours for the game introscreen, game colours (border), game over screen and game completion screen. 'E' exits.

T – Test game

This enables you to play the game with your current settings. Key 'Z' skips through the levels until completion. Be careful how you have set things up in case you get stuck, for although I have not managed this in recent versions it may be possible. It still is a good idea to save a snapshot before testing.

S – Save game

For when you have finished editing your game. A very good time to save an emulator snapshot in case there are problems.

This exits to BASIC so you can save the code to a TAP file (*or even real tape if you're hardcore*). The game code occupies memory addresses 32512 to 53662. On return to BASIC you need to type SAVE 32512,21150 & save to an inserted tape. The game entry address is 34051, so to start the game in this code you type RANDOMIZE USR 34051.

If you want to return to the game editor from BASIC you type CLEAR 26999, then RANDOMIZE USR 27000.

Game introscreen

The default game intro screen uses a combination of:

- Game title, as entered in option 8 game text
- An enlarged picture, which is block 31
- 256 characters of text, as entered in option 'I' - introscreen text.

Introscreen music

There is a large amount of spare memory (~11k) above the game. Any music can be stored here.

To activate introscreen music (example uses music stored at 64000):

- Load your music starting at memory address 64000 (though you can go as low as 53662).
- Poke memory address 34117 with the music start memory address.
- For example if your music is at 64000 you poke 34117,0 and 34118,250 (250 x 256 + 0 = 64000)

Notes

- At the start of the game all flags are set OFF
- At the game start gravity is set OFF, you can insert blocks to turn on at the game start
- The introscreen enlarged picture uses block # 31

Flip screen games

If you want to create a flip screen game (as opposed to a 'beat the screen' level based game you need to: In the event manager set 'when player meets screen edge' to NEXT SCREEN and in Variables set the number of levels to 1 (there may be occasions you do not want to do this).

Do not insert the player sprite into the map, instead POKE 34138,player xcoordinate (0-31), POKE 34143, player ycoordinate (0-21) which will be the position the player starts the game at.

Redundant CALLS

This is mainly for assembly programmers. There are a few redundant CALLs to 38094 (which is just a RET instruction) through the code that can be overwritten for you own purposes.

34051 CALL 38099 ;The first thing done on loading

34114 CALL 38099 ;Just after the introscreen is drawn

34117 CALL 38099 ;Just after the one above, and before the gamestart keyread. This a good place to CALL a music routine.

34286 CALL 38099 ;The following two CALLs are in the middle of the main game loop

34289 CALL 38099 ;

34389 CALL 38099 ;Just before the game over screen is drawn

34421 CALL 38099 ;Just after the game over screen key press

34425 CALL 38099 ;When a level is completed

34458 CALL 38099 ;Just before the game completed screen is drawn

34490 CALL 38099 ;Just after the game completed keypress & return to restart

POKES

There are a number of little changes that can be made with pokes, which can be done at any time (i.e. it is safe to do so while the editor is active).

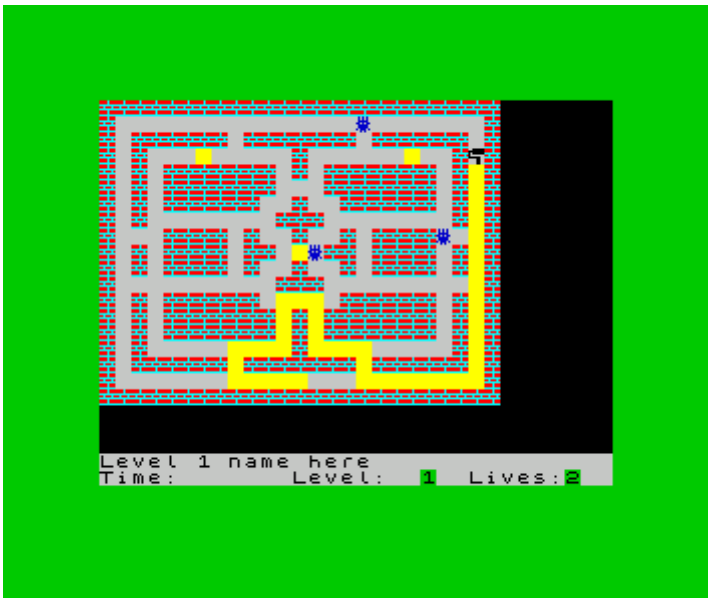
- POKE 34138,xcor (number 0 to 31) Player start X coordinate (for multiscreen game)
- POKE 34143,ycor (number 0 to 21) Player start Y coordinate (for multiscreen game)
- POKE 35181,n poke a different colour in for the 'red screen' effect
- POKE 39326,255 gravity ON (default is off)
- POKE 39327,n gravity strength, 3,7,15,31 are the only workable numbers higher = stronger
- POKE 38237,n block # used to delete a 'dec counter 1' block, default is 1
- POKE 38380,n block # used to delete a 'dec counter 2' block, default is 2
- POKE 38393,n block # used to delete a 'dec counter 3' block, default is 3

- POKE 39312,255 Makes a ‘tick’ noise every time the player moves.

SOME EXAMPLES

*The following downloadable examples use an old version of C.G.D, so are best used for inspiration and not as templates.

EXAMPLE 1: A SIMPLE PAINTER GAME



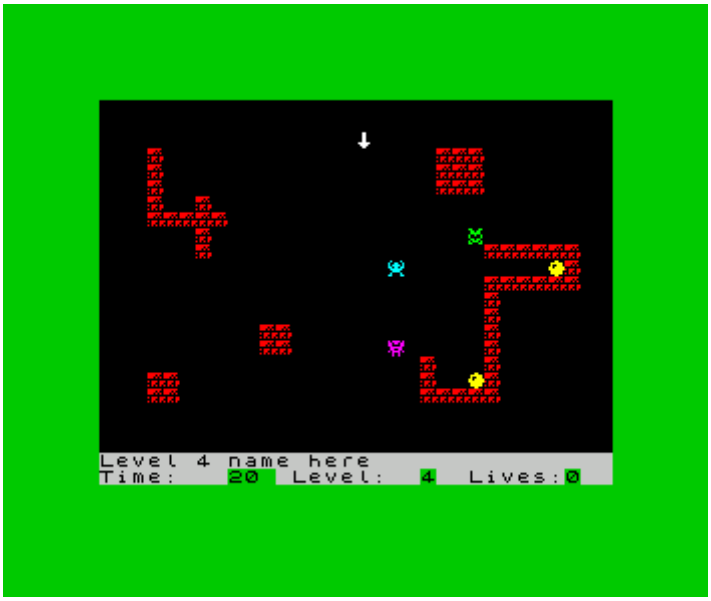
1. In the sprite designer create your player sprite right/left/up/down graphics in UDGs 0,1,2,3. In UDG 0 select ‘Pacman style keys’. I find speed 8 most suitable but see what you like. Create your baddy graphic in UDG 4, in this example I chose speed 8, behaviour – diagonal patrol and kill player. Have a play with various behaviours and you will see some work better depending how open the maze is. Don’t forget you can have a lot more baddy sprite types than the one in this example.
2. Now the block designer, create the following:
    - Block 0 – background block, this block is used to delete the sprites from the map when the screen is first drawn. Property: do nothing for both.
    - Block 1 – the ‘painted’ block. Property: do nothing for both.
    - Block 6 – the ‘unpainted’ block. This can be any block but I find it best keeping 1,2,3,4,5, free as they are used in some other specialist block types you may want to use later. **Property: dec counter 1** for player, do nothing for baddy. So every time the player lands on a ‘dec counter 1’ block it is deleted with block type 1 and counter 1 is decreased by 1.
    - Block 7 – the solid block. Select ‘solid block’ for both player and baddy.
3. Go to Item Counters, and press ‘B’, this automatically sets the counter to a particular block number. Set counter 1 to the ‘unpainted’ block. So every time the screen is drawn the

program counts the number of block 6 and sets counter 1 to that number. Don't worry about counters 2 & 3 for this example.

4. Go to the Event Manager, set screen complete when: to 'Counter 1 zero'.
5. Go to Variables. For this example set the number of levels to 1.
6. Now create your map. This example uses a small simple maze. Remember the counter is set to the 'unpainted block' so ensure you are able to collect ALL the 'unpainted' blocks on the screen. Find the player sprite by scrolling through the UDGs and insert it where you want it to start, and do the same for as many baddies as you want (max 11).
7. Test your masterpiece. This one is bloody hard!
8. Here's a link to the z80: https://docs.google.com/file/d/0ByxiMYbPnlUdV2R4eTJuQVpVWm8/edit?usp=sharing

EXAMPLE 2: A SIMPLE FLIP SCREEN GAME

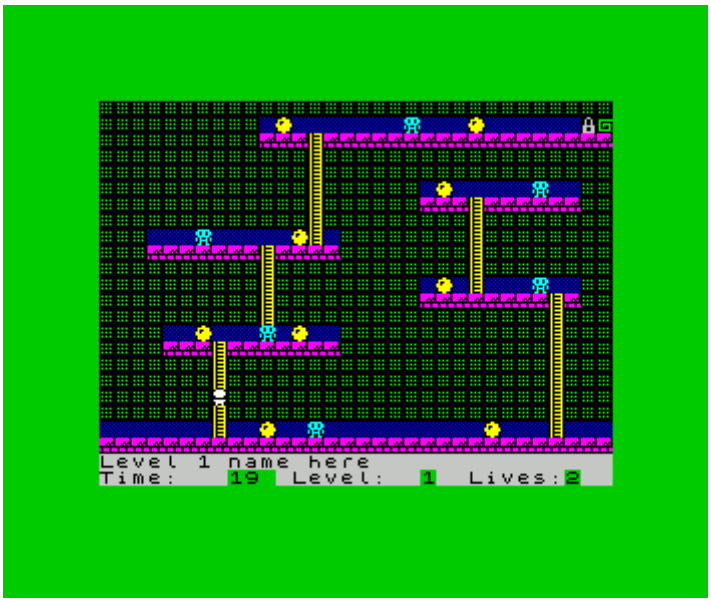
In this example you collect a total of 18 yellow coins over the 9 screens.



1. Create your player sprite and give it the properties you desire. Ditto baddies.
2. Create blocks, in this simple example there is just one object to collect and it is a 'dec counter 1' block.
3. In the Events Manager change 'when player meets screen edge' to 'next screen. In this example I am also setting 'Screen complete when' to 'counter 1 zero'. So when all the coins are collected the game finishes. In this case screen actually means whole game.
4. Item counters, we must select the 'N' option here. As there are two coins per screen and there are nine screens, we set counter 1 to 18. It is best to change the number of lives to 0, it does not have to be this but the default restart resets the counters which will probably upset your game, there is a way around it with pokes, see the pokes section for this.
5. In variables change the number of levels to 1.
6. Insert your blocks and baddy sprites into the 9 screens. Do not insert a player sprite, as if you do this when you move to that screen that is the position you will appear in. The default start position is at x,y 0,0. If you want a different start position you can POKE the value in:
    - POKE 34138,xcor (number 0 to 31)
    - POKE 34143,ycor (number 0 to 21)
7. Give it a test (another difficult one with just one life!) & think about how many things you can change here!
8. A z80 can be downloaded here: https://docs.google.com/file/d/0ByxiMYbPnlUdYVl1ekhSTGpNWWc/edit?usp=sharing

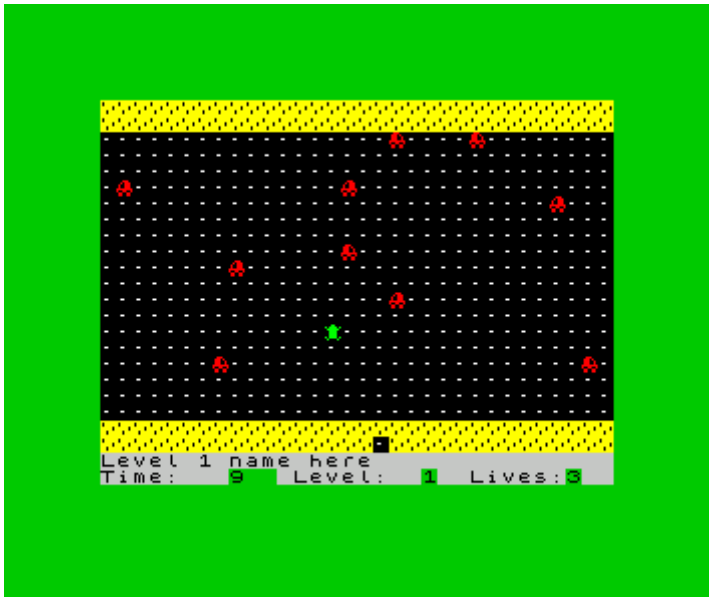
EXAMPLE 3: A SIMPLE PLATFORM GAME

Platform games are not best suited to CGD but a decent enough attempt can be made. This example is a simple platform and ladders, collect the coins and go to the exit. There are no tricks here, the ladders are not even a special block, it's mostly about the correct placing of solid blocks in the map and being aware which blocks will delete collectable objects and sprites.



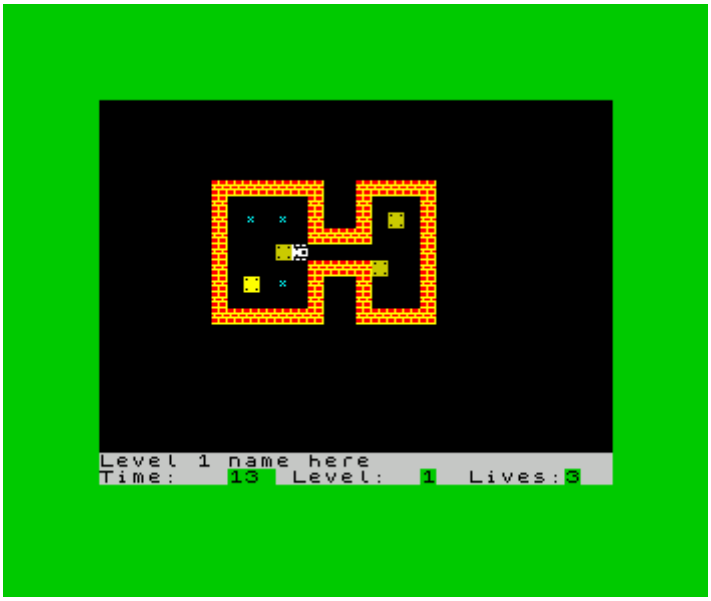
1. Design your player and baddy sprites. In this example I find multiple key press is the best key read.
2. Blocks: Most of the map is actually composed of solid blocks to restrict the sprites to moving in corridors. In this example note how the green background and magenta bricks are the solid blocks, and all the others can be moved through. The level is finished with a 'level skip' block at the top right, locked behind a 'Counter 1<>0: locked' block. This means the padlock block is locked until all the coins are collected. Ensure blocks 0,1,2,3 are <u>not</u> solid blocks, as these blocks are used for sprite and counter deletion.
3. In Item counters I selected 'B', set counters to blocks, and set counter 1 to the coin block. This is not essential, e.g. you can use option 'N' and set it to a particular number, but it means you must have the same number of collectable objects on every screen.
4. Give it a test and just think how many ways this can be varied.
5. The z80 is available here: https://docs.google.com/file/d/0ByxiMYbPnlUdMVpfdzdDekk3RzA/edit?usp=sharing

EXAMPLE 4: A SIMPLE FROGGER GAME



1. Create your sprites. In this example I went for the frog at speed 6 with Pacman style keys. Create as many vehicles as you fancy with varying speeds. This is where you use type: right only, and left only. In this simple example, just 2 cars, speed 8, one left and one right.
2. Blocks. Block 0 is the road block, property 'do nothing'. Block 1 is pavement, do nothing for the player, solid block for the baddy (this is to stop the cars randomly appearing on the top or bottom 2 rows).
3. Map. A simple layout of 2 rows of grass at the top and bottom, with the rest as road. Insert the frog at the bottom and space 10 car sprites (5 left, 5 right) through the road.
4. Event Manager. Change 'Screen complete when:' to player at top. Change 'Screen fails when:' to timer zero. Change 'When baddy meets screen edge:' to random wrap.
5. Variables, in this example change the number of levels to 1 and timer to 10.
6. POKE 39312,255. This makes a pleasing 'tick' noise every move.
7. Here is the z80, just think how many ways it can be changed: https://docs.google.com/file/d/0ByxiMYbPnlUdbGI1Smo2WmJoZUE/edit?usp=sharing

EXAMPLE 5: A SIMPLE SOKOBAN(-ish) GAME



This is mainly an example of how the push 'n' lock block works, it's not exactly like Sokoban as once it reaches the target it locks in position. Before attempting this please have a look at the properties of the 'push 'n' lock' block in the design blocks section of this document.

1. Create a player sprite, there are no baddy sprites in this example. I favour speed 8 and single key press.
2. Blocks. Block 0 is background colour and 'do nothing' (and the only block a push block can be pushed through), block 4 is the target block (usually leave as do nothing), block 4 turns into block 5 when the push block reaches its target. Block 6 is the 'push n lock' block, but it can be any block number. Remember when the 'push n lock' block reaches its target counter 3 is decreased.
3. Event Manager. Set 'screen complete when' to 'counter 3 zero'. This means when the last block is pushed into place the level is complete.
4. Item counters. Select the 'B' option, and set counter **3** to block 4, the target block. Using the 'B' option means you can have a different number of targets per level.
5. Variables. In this example the number of levels is set to 1.
6. Create your map. This example is not a particularly challenging puzzle.
7. Test and think about the possibilities! (There is also the push block and the push 'n' squash block which are less specialist). Here's the z80: https://docs.google.com/file/d/0ByxiMYbPnlUdSWNSejEyQzJ0dWs/edit?usp=sharing

Saving your game to TAP (or even real tape!)

Once you have finished your masterpiece you will want to put it all together, possibly with a loading screen. Here's one way of doing it:

1. Firstly save the game code. Press S (twice) in the designer and it will return to BASIC, insert a tape (eg in the emulator ZX SPIN it is in Recording>Tape Recording > Insert tape for saving. Find a TAP file (you can copy & overwrite an already existing one) and click save. The tape is now in the virtual recorder and you are ready to continue. Type SAVE "gamename" CODE 32512,21050 and press enter, the message 'Start tape then press any key will appear. Press enter and it will automatically save to tape. In Recording>TapeRecording, select eject save tape. Your game code is now safe.
2. Loading screens, if you want to do one ZX-Paintbrush (http://www.zxmodules.de/) is the best tool I have encountered for this. Create your amazing loading screen and save it as a TAP file.
3. Now to put it all together. Firstly insert a new tape (as in step 1).
4. Firstly we need a BASIC loader, this will autorun, load the loading screen and then the game code, then autostart the game. Here is how I do it, type the following:

10 CLEAR 32511:BORDER 0:LOAD ""SCREEN$:POKE 23739,111:LOAD""CODE:PAUSE 0:RANDOMIZE USR 34051

The clear instruction is important to stop any basic commands overwriting the game code. I chose border 0 because this is the same as the loading screen background. Load""Screen$ places your loading screen *on* the screen. POKE 23739,111 stops the 'Bytes' message overwriting your loading screen when the game code loads. Load""code loads the game code. Pause 0 is a very very long pause (or wait for keypress) so you can appreciate the loading screen before the game starts. Randomize USR 34051 is the entry address for the game, and starts it for us.

5. Save the above loader to tape with SAVE "name" LINE 1. The 'line 1' is important as it means you don't have to type RUN – it will autostart.
6. Now the loading screen. Type clear 32767 and press enter. Type LOAD""CODE 32768 and load your loading screen TAP file. The screen is now loaded from memory address 32768 (*this is better than loading it to the screen and saving it, as you don't lose the bottom border bit this way*). Type SAVE "screenname" CODE 32768,6912 and press enter. This saves your loading screen code as the second block on your tape.
7. Now for the game code. Type CLEAR 32511 and type LOAD ""CODE, then load the TAP file of your game. Now type SAVE "gamename"CODE 32512,21150 and press enter.
8. Eject the save tape – you've finished. Give it a test & hopefully it works!

Please enjoy CGD responsibly.