

Chain Reaction

Critical Mass

Blobs may appear to be totally harmless objects. They just sit around on a square playing field, looking shiny and colorful and explode from time to time. They only exploded when they reach a critical mass. **BY CHRISTIAN PERLE**

A program called *Chain Reaction*? Although it may sound like a virtual atomic reactor simulation, in fact *Chain Reaction* is a strategy game for two to six players. Written by Lee Haywood, *Chain Reaction* distributes the blobs in each player's color onto any free square or it increases the mass of its own blob conglomerations.

If a group of blobs reaches the total mass of the blobs in the surrounding squares, it explodes and spreads to the surrounding squares. If these squares were occupied by blobs of different colors before the explosion, they now assume the color of the exploding blob group.

If one of the neighboring blobs reaches critical mass on account of the explosion, it will also explode, and this can lead to extremely potent chain reactions. The game ends when all of the blobs are the same color. With this in mind, the aim of the game is to place your blobs strategically to recolor as many of your opponent's blobs as you can.

Do-It-Yourself

Chain Reaction is not available as a binary, and so you will need to compile the game. To do so, you will need various **libraries** – specifically **SDL** –



Fritz von Beust, vifpfx.com

and the appropriate development packages with the **header files**. These are:

- libSDL and libSDL-dev
- libSDL-image and libSDL-image-dev
- libSDL-mixer and libSDL-mixer-dev.

Package names can vary depending on your distribution. You might like to use the package search tool provided by your distribution!

The Chain Reaction source code archive is available at <http://www.deth.dsl.pipex.com/reaction.html> or on the subscription CD. The archive includes the *rcinst.sh* script that automates the installation steps. You will need to copy this file and *reaction-1.28.src.tgz* to a directory, before launching the installation by typing:

```
sh rcinst.sh
```

If everything works out okay, you will find the new software ready to run in the

/usr/local/bin and */usr/local/lib/reaction* directories.

Two or More

You can type *reaction &* in a **terminal emulation** to launch the game in its default two-player mode. To add more players simply click on a blank space in the score list on the right of the playing field.

The game starts with player one placing a blob in a square on the playing field. Figure 1 shows a game between two players. It is the blue player's turn, and she needs to decide whether to attack the red player's area (top left) or

OUT OF THE BOX

There are thousands of tools and utilities for Linux. "Out of the box" takes a pick of the bunch and each month suggests a little program, which we feel is either absolutely indispensable or unduly ignored.

THE AUTHOR

Christian Perle currently works as a developer at secunet Security Networks AG.

Christian discovered Linux in 1996, after playing around with the Sinclair ZX 81, Atari ST and finally IBM PC. When not hacking Linux stuff he can often be found playing guitar and "Magic: The Gathering".





Figure 1: Chain Reaction in Action



Figure 2: Has red trapped magenta?

defend her own area (bottom right). As red does not pose an immediate threat, you can assume that blue will choose the first option.

Trapped

Even though direct neighboring blobs may not reach critical mass when an explosion occurs, there are often more subtle ways of achieving this. You can arrange your own blobs to surround other squares and increase their critical mass from several angles, thus extending the chain.

In Figure 2, bottom left, red has surrounded the magenta colored blobs. However, it is magenta's turn, and she can foil this ploy. Before red's next turn blue might also pose a threat in the top left corner.

As the total mass distributed amongst the squares continually grows throughout the game, the chain reactions towards the end become increasingly severe. Shortly before the end of the game, major shares of the playing area can shift quickly, allowing an outsider to win a game with a single well-planned explosion.

When playing *Chain Reaction*, take care not to leave too big a gap in your blob area, as your opponents can easily exploit it. As the corner squares only have two neighbors, the danger of explosions is extremely high – if an opponent has control over a corner (see Figure 2 lower right), you need to act fast, as the critical mass will be reached in the next round.

Terraforming

If a square playing field gets boring, you can use the integrated game editor to create your own “worlds”. Before you start playing, simply press the right mouse button to remove or add individual squares. Isolated squares are not permitted, as they have no neighbors and thus cannot be re-colored.

If you are happy with the playing field you have created, you can click on the *Save* button at the bottom of the windows to store the game to the *saved-game* file in the current directory. This function not only saves the game, but also the current positions and previous moves. The *Undo* and *Redo* buttons

allow you to re-construct any moves previously made.

Figure 3 shows an edited playing field. Some corner squares only have a single neighbor. Any blobs placed on these fields explode immediately and attack their neighbors' mass. So these squares remain permanently empty and pose a constant threat of attack. This also forces you to rethink your tactics; you cannot concentrate on the corners, as it is impossible to defend them. ■



Figure 3: The playing field editor livens up the game

GLOSSARY

Library: A “library” contains a collection of useful C functions for specific purposes, such as *libXt* that provides functions for X Window programming. Libraries are often used by multiple programs, that is they are shared.

SDL: The “Simple Directmedia Layer” library allows both graphic and sound output independent of the hardware and operating system.

Header files: Header files (also known as “include files”) list the functions and parameters available in a library. The C compiler requires this information when it translates a program. Most distributions add the *dev* or *devel* extension to the header packages for a given library.

Terminal emulation: Years ago, mainframe users used to enter commands on so-called

terminals (ttys). Programs that emulate this kind of device (adapted to modern computer and operating system standards) have inherited the name. On Linux, a terminal emulation refers to displaying a virtual console in VGA text mode or a separate X Window program for command-line access, such as *xterm*, *console* or *gnome-terminal*.