

Dedoimedo

A place to learn a lot about a lot!

5057 roger

Computer games

Life topics

Hillbilly physics

Greatest sites

3D art

Model planes

GParted partitioning software - Full tutorial

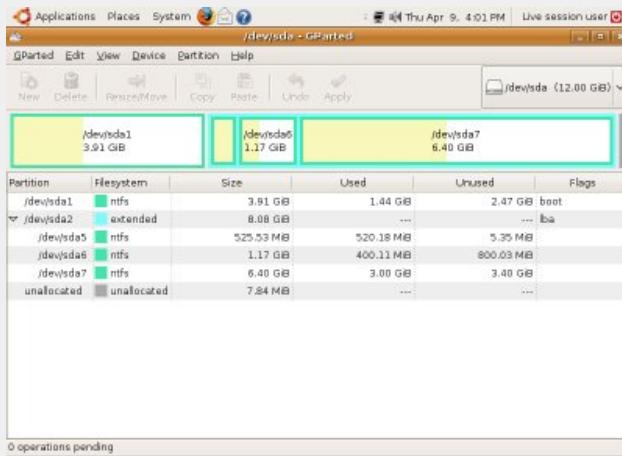
Updated: April 25, 2009

No matter which operating system you want to have installed on your machine, there's one thing they all have in common: partitions. These are logical pieces of your actual hard disk space, defining the size and the file system format for the operating systems and data that are going to be placed on them.

Being able to control the partitions is one of the most basic, most important aspects of mastering your operating system. If you have the knowledge and confidence to manipulate the layout, create it, change it or delete it, you can adapt your hardware to your varying needs, without having to blindly rely on default setups defined by vendors or other people.

After completing this tutorial, you will have learned how to interpret the partitioning dictionary, what the strange symbols, letters and numbers mean. You will have learned how to create partitions or edit existing ones, including changing their filesystem, size, type, or structure. Most importantly, you will have learned how to read existing setups, be they Linux, Windows or something else.

Never again you shall fear using partitioning software or installing operating systems on hard disks already containing data. After this tutorial, you will know how to handle partitioning with genuine knowledge. So let us begin.



Advertise!

Would you like to advertise your product/site on Dedoimedo?

[Read more](#)

Donate to Dedoimedo!

Do you want to help me take early retirement? How about donating some dinero to Dedoimedo?

[Read more](#)



Table of contents:

1. [GParted - Introduction](#)
2. [How to use GParted?](#)
 - A. [Practical examples](#)

[3. Partitioning dictionary](#)

- A. [Windows uses drive letters](#)
- B. [Linux notation is different](#)
- C. [Partition types](#)
- D. [Important thing to pay attention to!](#)
- E. [Exceptions](#)
- F. [What to install where?](#)

[4. General partitioning recommendations](#)

[5. Using GParted - Understanding the software](#)

- A. [Core functions](#)

[6. GParted - real life examples](#)

- A. [Identifying the right device](#)
- B. [Task 1: Resize partition](#)
- C. [Task 2: Create new partition](#)
- D. [Task 3: Delete partition](#)
- E. [Task 4: Create Partition Table](#)
- F. [Task 5: Create only Extended partition](#)
- G. [Task 6: Move partition](#)
- H. [Task 7: Check & repair filesystem](#)

[7. Flags](#)

[8. GParted capabilities](#)

[9. Advanced tasks](#)

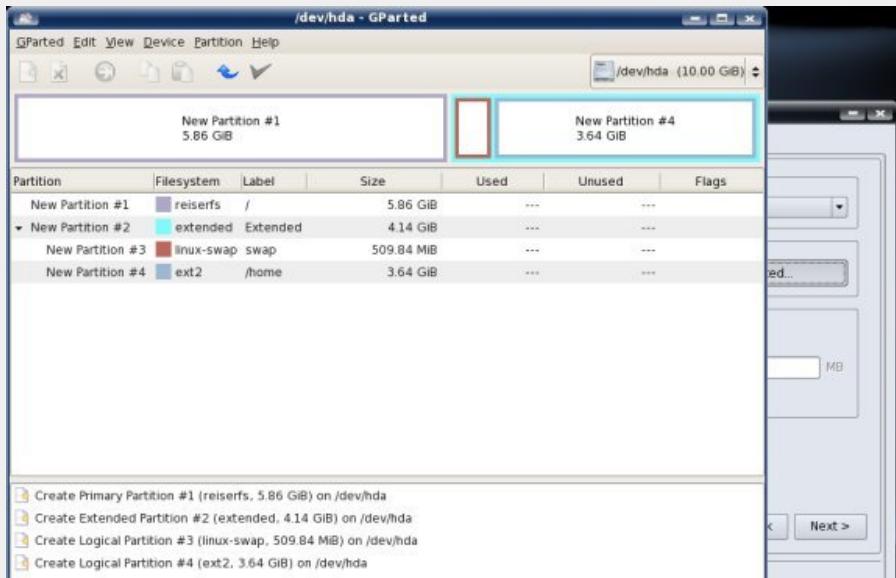
- A. [Change the Inode size](#)

[10. Recommended reading material](#)

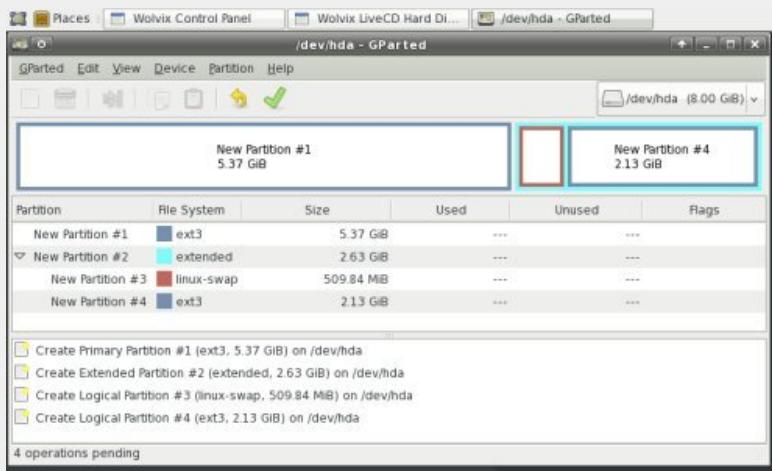
[11. Conclusion](#)

GParted - Introduction

GParted is one of the most popular partitioning software. It comes included with most modern Linux distributions. It also ships in a large number of dedicated rescue & recovery distributions. To name a few distributions that come with GParted: Ubuntu, Linux Mint, PCLinuxOS, Wolvix, and others. You can read tutorials and reviews for these in my [Software](#) section. GParted is a graphical software, so it is well suited for modern use, including less knowledgeable users. Here's what GParted looks like:



Or like this:



Basically, the decorations may vary, but it will be same software underneath. Do not worry about what you see, either. We will soon learn in great detail how to interpret GParted results.

How to use GParted?

GParted can be used in two ways: while booted in an operating system or from a live CD. The recommended way of using GParted is from the live environment. Why, you ask? This is because partitioning operations need to be done on hard disks when they are **not** in use, to avoid data corruption. Partitions that are in use cannot be modified. They are locked by the operating system that uses them.

In technical terms, partitioning can be done only when the hard disk partitions are **unmounted**. If disks are empty and contain no operating system whatsoever, it does not matter anyway, because the only way you can access the system is from a live environment.

As a rule of thumb, it is always the best idea to handle partitioning from live CD environment. Not surprisingly, almost every single modern Linux distro ships as a bootable live CD. Not only does this allow you to get a first impression of the operating system and check hardware compatibility before deciding whether to commit the distro to hard disk, it also allows you to perform maintenance operations from the live environment.

Nevertheless, you can still use partitioning software against NON-system partition, that is partitions that the operating system is not installed on, and which, on demand can be unmounted. This is true for Windows and Linux alike. And just about any operating system in the world. I may have confused you, so let's recap the uses of partitioning software:

- Partitioning software cannot be used on partitions that are used (mounted) by an operating system.
- Partitioning software can be used on system partitions only when booted in a live CD environment.
- Partitioning software can be used on data partitions or empty, non-system disks while booted in either local, installed operating systems or from a live CD environment.

Practical examples

Example 1: Let's say you have Windows installed on drive C: and you have data (movies) on drive D:. Drive D: is formatted with FAT32 and you would like to convert it to NTFS. You **can** do this without booting into a live CD session. Since the system uses C: drive, there is no problem unmounting drive D: and changing it as necessary.

Example 2: Let's say you want to resize the same drive C: as above. You cannot do that while booted in Windows, because the system uses the drive. You will have to boot into a live CD environment, Linux or Windows-based and perform the partitioning changes from there.

Example 3: You are dual booting Windows and Linux. Currently, you are booted into your Linux. You wish to change your Windows drive C:. Even though drive C: is the Windows system partition, when you're booted in Linux, it is not active. Therefore, it is just like the data partition we worked on in

example 1. This is very similar to working from live environment. However, in a live environment, you could also choose to work on the Linux root (/) partition as well, whereas when booted in the Linux operating system residing on the disk, you can only work on other, non-system partitions.

Example 4: The same dual-boot system, only this time you're in Windows. In general, Windows cannot see Linux partitions, although there is [software](#) that can overcome this limitation. Assuming that you can see the Linux partitions, you can change their partitioning layout, including the Linux root partition, because it is currently not in use.

I hope these examples help clarify the situation somewhat. The things are quite simple. Partitions used by the system cannot be edited as long as they are used. Data partitions can be edited in vivo. Whatever you do, it is prudent to think twice and [backup](#) any critical data before making changes. Now, let's talk about the notation.

Partitioning dictionary

Let's now try to understand how GParted sees hard disks and marks them. If you're a Windows user or have just started with Linux, the notation may be unfamiliar to you. Not to worry, we will have it explained to the latest detail:

Windows uses drive letters

In Windows, users are accustomed to referring to their partitions as drives, like C:, D: etc. This is somewhat misleading, because these letters in fact refer to partitions rather than actual drives. If you have a single drive (only C:), then the term partition and drive are synonymous in this case, because a single partition spans the entire size of the hard disk.

However, if you have more than a single drive letter in your My Computer, this means you have several partitions (and maybe even several physical hard disk drives). It is important to remember this.

Linux notation is different

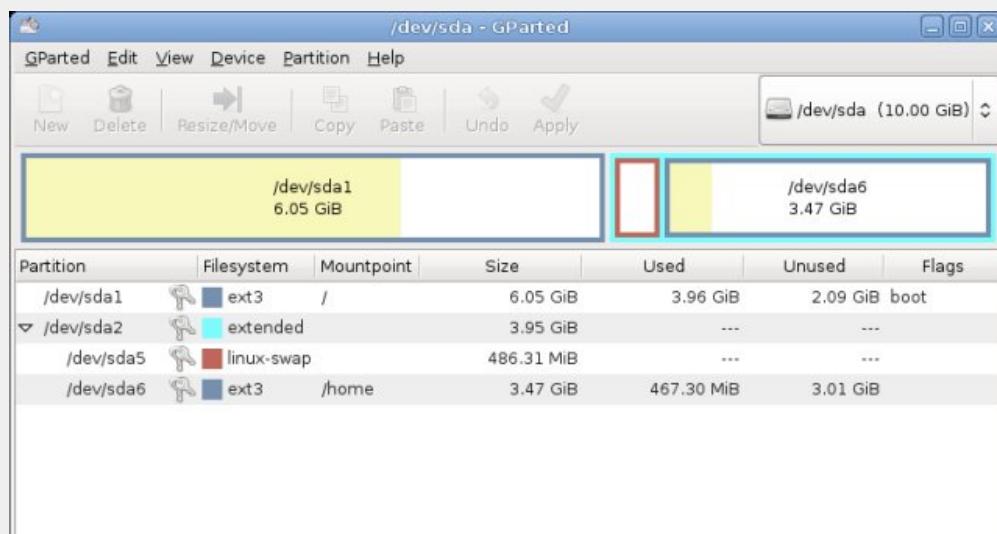
I have explaining the Linux disk notation in many other articles, but for the completeness' sake, I will do it one more time.

Hard drives in Linux are marked by three letters:

- IDE drives are marked **hdX**, where X is one of the four letters a-d. **hda** is the primary master, **hdb** is the primary slave, **hdc** is the secondary master, and **hdd** is the secondary slave.
- SCSI / SATA drives are marked by **sdX**, where X is any which letter.

Partitions are marked by a number after any three letter combination:

For example, **sdb1** is the first partition on the second SCSI / SATA drive. **s** - SCSI/SATA, **d** - drive, **b** - second drive, **1** - first partition. **hdc3** is the third partition on the IDE secondary master. Here's a screenshot of the partitioning layout on one of my machines:



0 operations pending

And here's what it looks like in text form:

```
roger@roger-desktop: ~
File Edit View Terminal Tabs Help
roger@roger-desktop:~$ sudo fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x0000b1806

Device Boot Start End Blocks Id System
/dev/sda1 * 1 790 6345643+ 83 Linux
/dev/sda2 791 1305 4136737+ 5 Extended
/dev/sda5 791 852 497983+ 82 Linux swap / Solaris
/dev/sda6 853 1305 3638691 83 Linux
roger@roger-desktop:~$
```

What do we see here?

Let's take a look at the first picture. Don't worry about using GParted, we'll get to it. What I want you to focus on are the color ribbon and the partition notations. As you can see, all partitions are marked with sdaX. This means we have a SCSI/SATA disk at hand. The numbers indicate the partition order. The second image shows the same information in text form.

There's more information to be had from this example, but we will talk about it later on. One thing I want to focus on is the sequence of numbers. You may have noticed we have sda1, sda2 and then sda5, but no sda3 or sda4 in between. For those unversed in the rules of partitioning, this can be confusing. This is why it is important to understand partition types.

Partition types

Partitions also have another important element: they can be **primary** or **logical**. Primary partitions are just that, a total of four of which can exist on any one hard disk. To reiterate, there can be only up to four primary partitions on a hard disk. If you have three hard disks on your machine, each one can still hold up to four primary partitions.

Logical partitions have been created to overcome the inherent numerical limitation of primary partitions. One of the primary partitions can be created as the **Extended** partition. This partition acts as a container for logical partitions. The total number of logical partitions you can create (and use) depends on the disk type and the operating system you're using. For all practical purposes, the number is beyond the needs of any user.

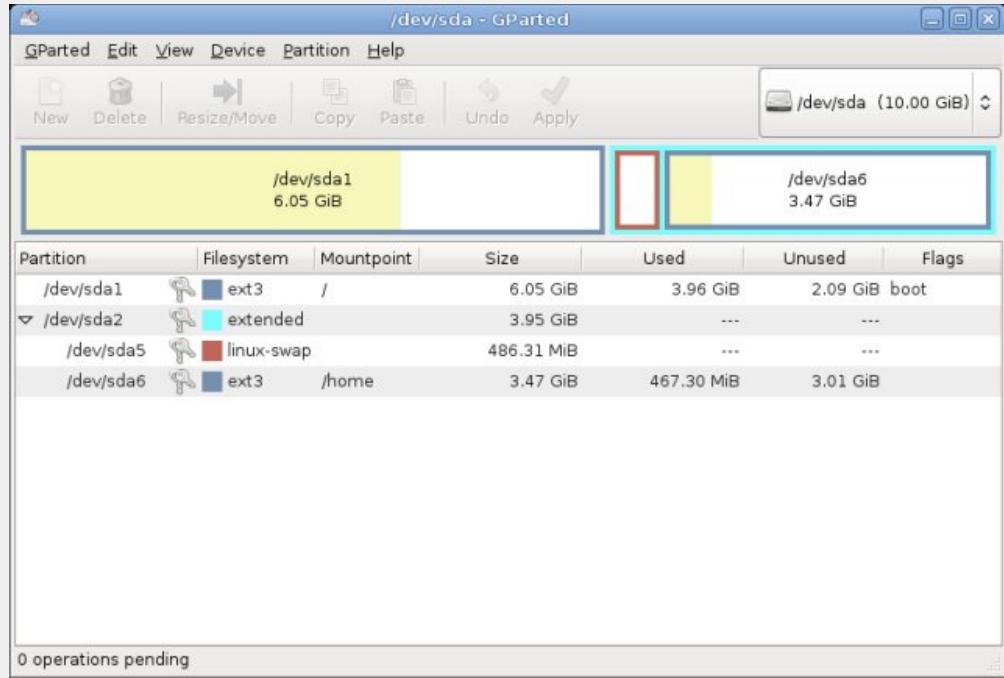
As you can see, we have up to four primary partitions and a de-facto unlimited number of logical ones. Notation-wise, the primary partitions will always be the first four, logical partitions will start with number **5**.

Therefore, when someone says **sda5**, it necessarily means we're talking about a logical partition. Similarly, any partition with a number equal or higher than 5 will always be a logical partition.

Important thing to pay attention to!

It is also **important** to understand that although sda5 is the fifth partition per se, there do not have to be four primary partitions on the system. There will be either one, the extended partition itself, which is the bare minimum, or more (up to four). Therefore, notation-wise, logical partitions begin with number 5. Physically, sda5 is the **FIRST** logical partition. Physically, it can be fifth, but it can also be anywhere between first or fifth.

Please remember this. This is very important! Why, you ask? Because if you use a visual tool for partitioning, like GParted, do NOT count the partitions visually!



We have seen this layout before; it was the sample layout we reviewed earlier.

It's a very good example, as the matter of fact. This is because, in this case, sda5 is the second partition on the system! sda1 is the primary partition that holds the root filesystem of the specific Linux operating system installed on the machine. sda2 is the extended partition, which contains sda5. So if we count from left to right, **sda1** is our **first** partition, **sda2** is the extended partition, but it is a container for all logical partitions, so we cannot include it in our visual count! Therefore, **sda5** is the **second** rectangle on the color ribbon!

I implore you to pay attention to this subtle fact! Never, ever blindly count partitions just based on their numbers. Always triple check that you're working on the right hard disk, on the right partition. And always backup data before making changes. Never edit partitions without a proven, tested recovery plan in place!

Exceptions

All of the examples mentioned above relate to single disk configurations. They do not take into account Redundant Arrays of Inexpensive Disks (RAID) or Logical Volume Manager (LVM). In this tutorial, we will not go into the management of these solutions too deeply, because they are inherently more complex.

However, I won't leave you without a solution - we will talk about RAID and LVM in a separate tutorial. For now, please accept my apologies and try to get by with just a brief introduction on "cross-disk" solutions.

RAID

RAID stands for Redundant Array of Inexpensive Disks. This is a solution where several physical hard disks (two or more) are governed by a unit called RAID controller, which turns them into a single, cohesive data storage block.

An example of a RAID configuration would be to take two hard disks, each 80GB in size, and RAID them into a single unit 160GB in size. Another example of RAID would be to take these two disks and write data to each, creating two identical copies of everything.

RAID controllers can be implemented in hardware, which makes the RAID completely transparent to the operating systems running on top of these disks, or it can be implemented in software, which is

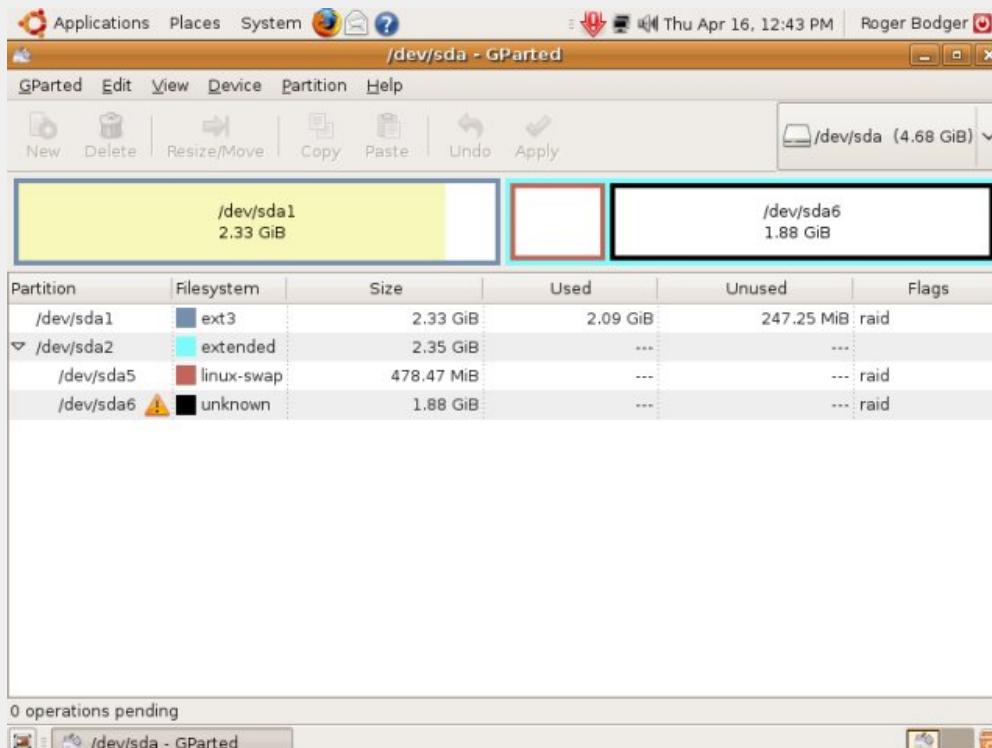
the operating system running on top of these disks, or it can be implemented in software, which is the case we are interested in.

There are quite a few RAID schemes, known by numbers and names, such as RAID 0, RAID 1, RAID 5, and others. You may also have heard of RAID striping and mirroring, which are names for RAID 0 and RAID 1, respectively. If you're interested, Wikipedia has a very nice [article](#) on the subject.

RAID is interesting, because we can no longer use physical disks and partitions as units of measure. Instead, we have a higher level of hierarchy instead, defining how the devices should be called. If you remember this important fact when setting up RAID, it will be much easier for you to understand the concept.

RAID devices in Linux are denoted by letters **md** followed by a single letter. For instance, md0, md1, md6, these are valid examples for RAID devices. There is no strict relation whatsoever between md devices and physical hard disks and their partitions.

For example, **md0** could be a RAID 0 device, spanning physical sda1 and sdb1 partitions. It could also be a RAID 1 device, spanning physical sda1 and sdb2 partitions. In both cases, the device name would remain the same, while the physical topography underneath would be different. Here's an example:



We can see that GParted does not display RAID (md) devices, but it does identify them. The RAID partitions are marked with the **raid** flag (more about those later).

One thing worth noting is that on **sda6**, GParted is unable to recognize the filesystem. This is because the RAID configured on that partition is such that sda6 does not provide all the information on the filesystem used, preventing GParted from properly classifying the partition. We're using RAID 0, known as **striping** on sda6 (and sdb6), which converts these two partitions into a single device. Therefore, each partition contains only half the information, hence deciding on what data is contained cannot be deducted from just looking at a single partition in the pair.

This should not bother you, as it's perfectly all right. However, you should remember that this can happen - and know what it means. We will talk about this in great detail in a dedicated tutorial. Another example, this time using the command-line utility **fdisk**, here's what a RAID layout might look like:

The screenshot shows a terminal window with the title "root@ubuntu: /". The command "root@ubuntu:/# fdisk -l" was run, displaying the following output:

```
Disk /dev/sda: 5033 MB, 5033164800 bytes
255 heads, 63 sectors/track, 611 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x0002fe30

Device Boot Start End Blocks Id System
```

```
/dev/sda1      1      319    2562336  fd  Linux raid autodetect
/dev/sda2    320     383    514080   82  Linux swap / Solaris
/dev/sda3    384     611   1831410   83  Linux
```

```
Disk /dev/sdb: 5033 MB, 5033164800 bytes
255 heads, 63 sectors/track, 611 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x000e8c89
```

```
Device Boot  Start    End    Blocks  Id  System
/dev/sdb1      1    319    2562336  fd  Linux raid autodetect
/dev/sdb2    320     383    514080   82  Linux swap / Solaris
/dev/sdb3    384     611   1831410   83  Linux
root@ubuntu:/#
```

Notice the Linux raid autodetect filesystem. This means that partitions **sda1** and **sdb1** might be used in a RAID configuration. What and how exactly, we will focus on that in a separate article. Another useful command for checking the status/presence of RAID devices on the system is the [/proc/mdstat](#) command:

```
Applications Places System Sun Apr 12, 6:33 PM Live session user
root@ubuntu:/# cat /proc/mdstat
Personalities : [raid1]
md1 : active raid1 sdb2[1] sda2[0]
      513984 blocks [2/2] [UU]

md2 : active raid1 sdb3[1] sda3[0]
      1831296 blocks [2/2] [UU]

md0 : active raid1 sdb1[1] sda1[0]
      2562240 blocks [2/2] [UU]

unused devices: <none>
root@ubuntu:/#
```

```
root@ubuntu:/
```

For example, on the system above, we have three RAID devices, md0-2, each containing a pair of devices in a Mirror configuration, also known as RAID 1. Again, do not get flustered if you find this short sub-section too technical. A separate tutorial will explain RAID in detail.

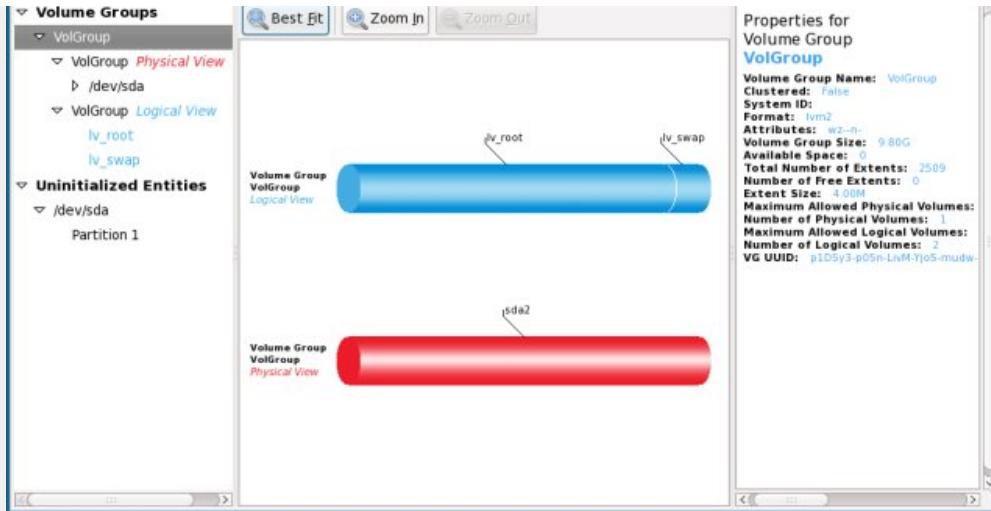
While GParted can identify RAID devices, it cannot create or fail them. To this end, you will have to use other utilities. For now, though, it is important that you understand what RAID is what it looks like, so you can properly identify the layout and change it accordingly if needed.

LVM

LVM is somewhat similar to RAID. However, it is different in being able to allocate any which bit of hard disk space into logical sub-groups, known as Volume Groups, each containing one or more Logical Volumes.

The easiest way to visualize LVM is as a space-restriction-free partitioning on top of an existing physical disk layout. In order words, no matter how many disks or partitions you have, you can ignore them and use a higher order of hierarchy known as logical volumes, managed by LVM.

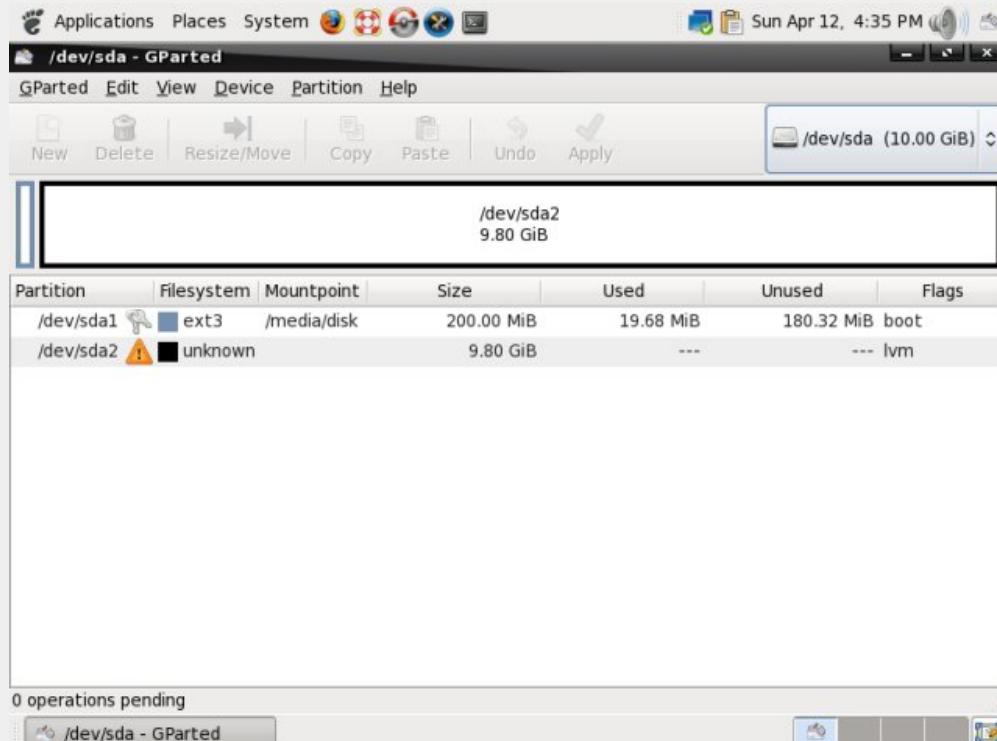
```
Logical Volume Management
File Tools View Help
```



Above, you can see an example from the default [Fedora 11](#) installation. Please take a look at the [Physical View](#) and [Logical View](#) separately. Let's try to understand what we see. The Physical View tells us our Volume Group sits on [sda2](#), a primary partition. What we do not see is [sda1](#), which in fact is a small [/boot](#) partition used to boot the system.

Logical View shows us what is contained inside each Volume Group, ignoring the actual physical devices. In our case, we have a single Volume Group, which contains two Logical Volumes, root and swap. For all practical purposes, we do not know or care what configuration exists underneath.

Our LVM takes 90% of hard disk space, but it could also take anywhere between 1% and 100% of any which hard disk and partition that physically exist. For example, if we had two hard disks on the system, LVM could take 54% of the first disk and 90% of the second. Furthermore, this arrangement could span any number of partitions. Here's what the same layout above looks like in GParted:



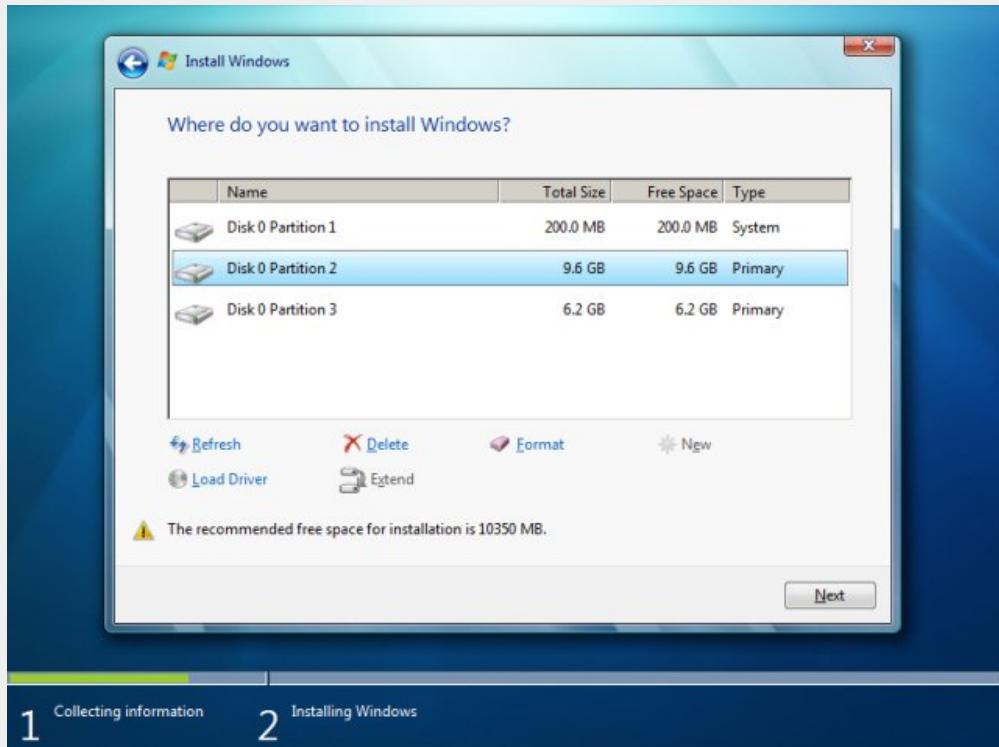
We have a small EXT3 partition that is used to boot the operating system. You can tell this by the [boot](#) flag. And then, we have an unknown filesystem on [sda2](#), which is our LVM; again notice the flag. The filesystem is unknown because the partition may contain several Groups, each with several Volumes, each with a different filesystem. So the question is, which of the possible choices should GParted choose.

LVM introduces a high degree of freedom and flexibility, allowing users to span physical limitations of individual partitions and/or drives. LVM uses a tricky notation. We won't discuss it in detail here. However, you should be aware of the facts. Like RAID, LVM devices have a special flag denoting them. Remember this when we review different types of partition flags later.

What to install where?

The limitation of only four primary partitions is critical when considering a future setup. It definitely forces us to carefully think through our installation needs and requirements. To make things worse, some operating systems REQUIRE that they be installed on primary partitions.

Windows is a good example. To have Windows (XP, Windows 7, etc) function properly, they must be installed on primary partitions. To make it even worse, the first primary partition. Take a look at my [Windows 7](#) review, including the partition. Windows 7 ungenerously grabbed no less than three primary partitions for itself!



BSD operating system flavors also like primary partitions. So does Solaris. Take this into consideration when planning multi-boot setups. Linux is far more flexible and can be installed on any partition. Because of this, it is always a good idea to use logical partitions for Linux, when you can, so you do not waste the precious few primary partitions.

General partitioning recommendations

OK, here's a brief summary on what we have learned so far:

- Windows and Linux uses different notation. Windows marks partitions with letters and calls them drives - not necessarily corresponding to *physical* drives. Linux uses three-letter and one-digit notation, beginning with h for IDE and s for SCSI/SATA drives. The third letter marks drive number, as seen by BIOS, with a-d for primary/secondary master/slave for IDE drives and unlimited numbers for SCSI/SATA drives, based on the controller limitations. The digit refers to partition numbers.
- Numbers 1-4 are used to denote primary partitions, one of which can be an extended partition, a container for logical partitions.
- Logical partitions will always be marked with number 5 and higher. Physically, logical partitions can be less than their actual number, depending on the number of primary partitions that exist on the system.
- Partitions are counted separately for each physical hard drive as recognized by the system. The exceptions are RAID and LVM configurations.

Now, useful tips to remember when playing with partitions:

- Windows requires primary partitions.
- BSD and Solaris also require primary partitions.

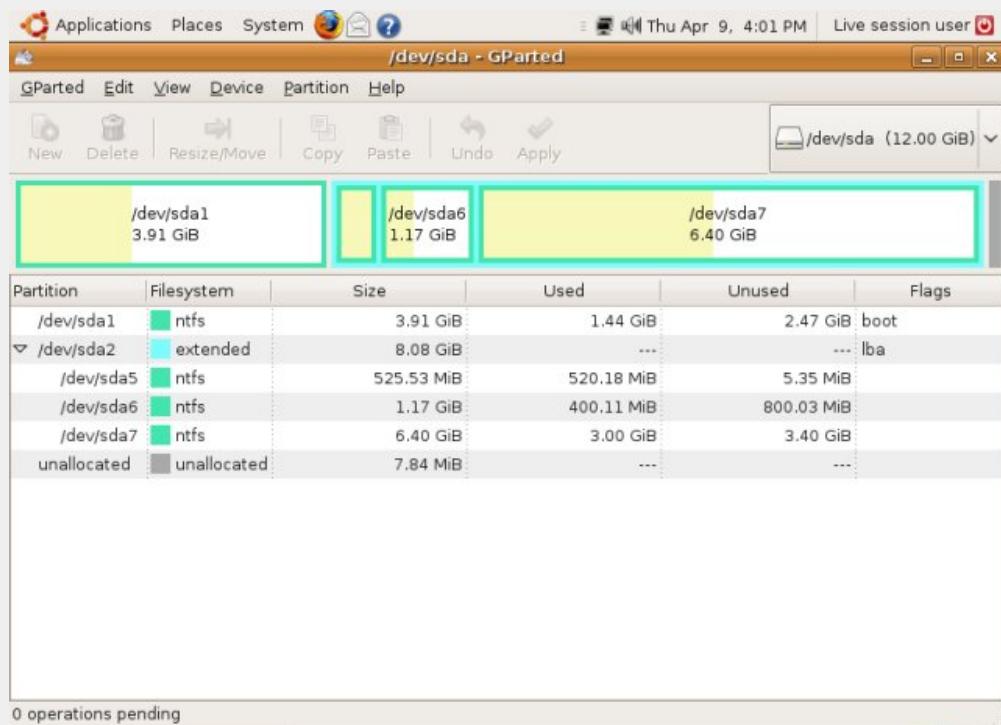
- Linux does not need primary partitions and can be installed on logical ones.
- Always install operating systems that require primary partitions first.
- Carefully think through your partitioning needs and create partitions before installing operating systems. Think seven steps and three years ahead and make sure you have enough room to grow. Scalability is an important factor. Make sure your partitions are neither too small nor too large.
- Do not forget size limitations for older file systems (like FAT32).

So, now we have a basic understanding of what to expect. Let's start using GParted and review real-life test cases.

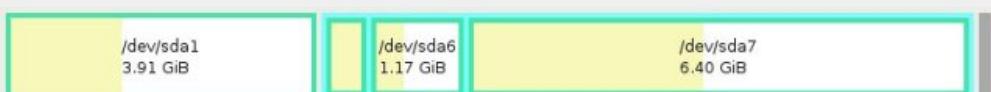
Using GParted - Understanding the software

The first thing to do is to launch the application. The exact location of the utility in the menus will vary from one distro to another. For instance, on Ubuntu, you will find GParted under **System > Administration > Partition Editor**.

Whether you're working in-vivo or from a live CD, you'll need administrative (root) privileges to work with partitions. Now, before we use GParted, let's make a quick look of its functions. When you launch GParted the first time, it will scan the existing devices on the machine and present a layout for each hard disk separately. It will open displaying the information for the first disk (as recognized by BIOS). Something like this:



Like most GUI tools, GParted has functions displayed both as buttons and entries in the File menu. This means you can perform every tasks in two different ways. Partition layout, if it exists, is displayed on a visual ribbon, with different colors marking different partitions and their filesystems. Free hard disk space will be marked in gray. Free spaces on existing partitions will be marked in white. Partition space filled with data will be marked in yellow, with the visual fill-up bar roughly corresponding to actual percentage taken.



The same information is also shown in the table form below the color bar. The **Partition** column will list all existing partitions on the particular device, starting with **/dev/** for device, followed by hdXY or sdXY notation, we already discussed.

The second column, **Filesystem** indicates the filesystem the partition uses, if any. Different filesystems are marked by different colors, so there are no mistakes. If a partition is in use by the system, there

will also be a key symbol displayed near the partition, indicating it is used (mounted) and that operations cannot be performed on it.

Partition	Filesystem	Mountpoint
/dev/sda1	ext3	/
/dev/sda2	extended	
/dev/sda5	linux-swap	
/dev/sda6	ext3	/home

The **Mountpoint** refers to a directory under the root (/) where you can access the data contained on the partition. Unlike Windows, which separates drives by their letter and treats each individually, all filesystems on Linux are mounted under a single tree, aptly called root. Even if you have network shares used by the system, they are accessed the same way as local files, by changing path into one of the directories or sub-directories. Thus, for instance, if you access /home, you will see all the data that is physically written on the /dev/sda6 partition.

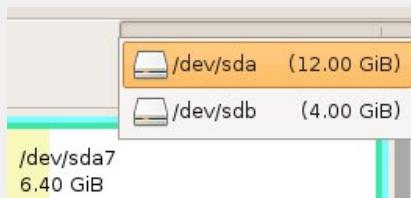
The **Extended** partition has no mountpoint, because it is not used directly. It's a container. swap is also special. It's similar to the Windows pagefile. **swap** is a piece of hard disk used by the system to swap between real and virtual memory, increasing the processing capacities on the expense of some performance loss. As such, swap is not used manually by users; it's treated as a raw device. Read to and write from swap is done on the partition level rather than via mountpoints and human-readable filesystems.

Size, Used and Unused are all part of the same equation - partition capacity. I believe they are self-explanatory.

Flags are interesting. In order to be able to understand what each partition does, operating systems use flags. One of these flags is the **boot** flag, which tells the system, be it Windows or Linux or any other, that the particular partition marked with the boot flag is the one where the operating system should use to boot. Another useful flag is **lba**, which stands for Logical Block Addressing; you can read more about LBA on [Wikipedia](#).

Unused	Flags
2.47 GiB	boot
---	lba
5.35 MiB	

I've mentioned earlier that by default, GParted displays the first device only. But what if you want to work on the second hard disk? Not to worry, switching it very easy. In the right corner above the color bar, there's a drop-down button, allowing you to change visible devices.



And the view will then switch to relevant device:

Partition	Filesystem	Size	Used	Unused	Flags
/dev/sdb1	ext3	4.00 GiB	104.37 MiB	3.90 GiB	

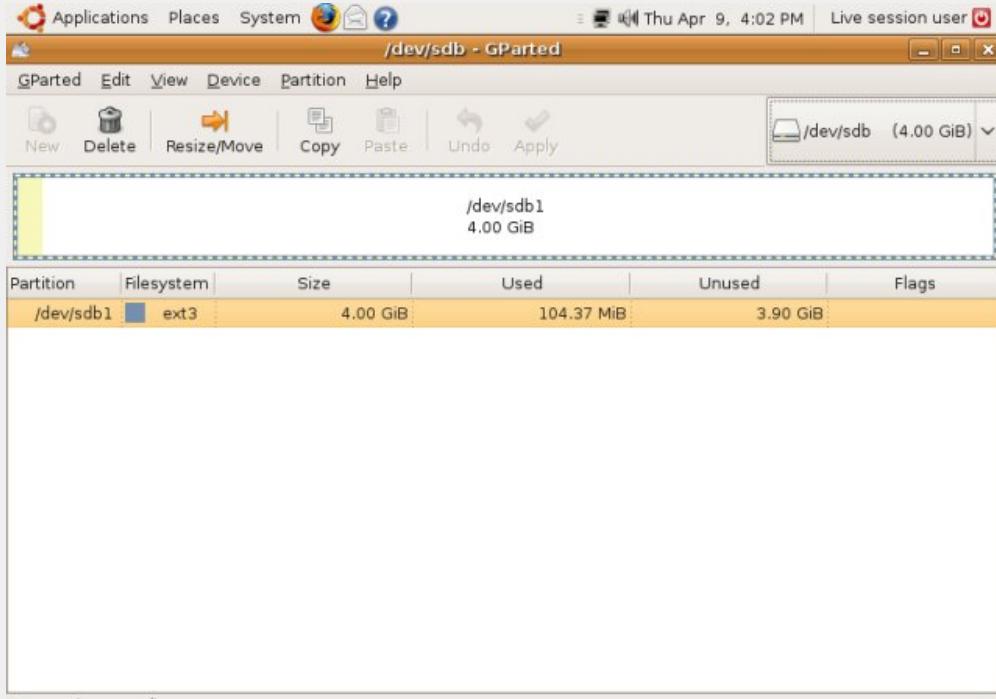
0 operations pending

/dev/sdb - GParted



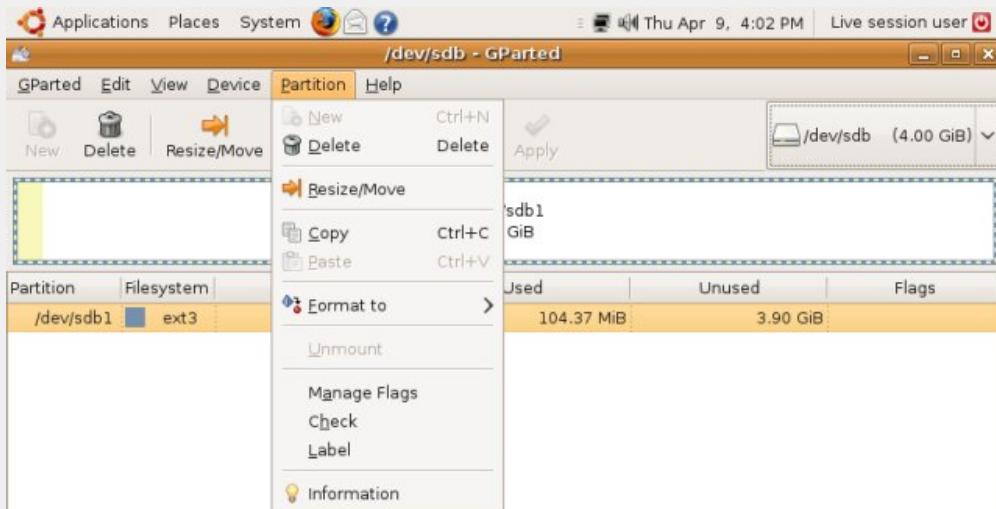
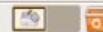
Core functions

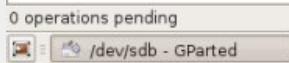
The core functions of GParted are the creation, resizing/moving, deletion, and formating of partitions. The usage is very simple: highlight the relevant empty space or an existing partition and perform the desired tasks. You can use the buttons or the menu. The buttons/functions will be grayed out until you choose the relevant bit of hard disk space to work on:



0 operations pending

/dev/sdb - GParted





Now, we're ready to start working.

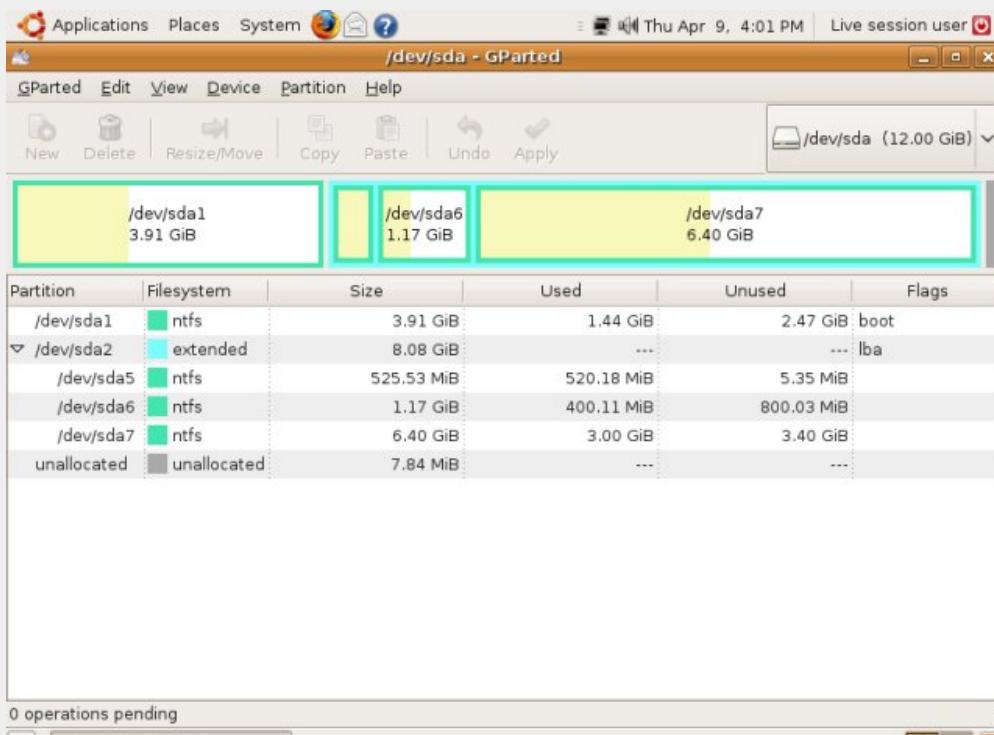
GParted - real life examples

Our test case is a machine with two SATA disks. On the first disk, we have Windows installed, with several data partitions. The second disk is currently occupied by a single Ext3 partition. This is an excellent example of a complex system that a new Linux user will face when trying to install the Linux for the first time. If the disk is empty, the choices are rather simple. But what about a disk already used, with critical data on it? Not to worry, we'll have it sorted out.

Identifying the right device

We know the notation, we're familiar with GParted GUI. Now, all we need is to decide what our target device will be. Let's see what we have:

First disk:



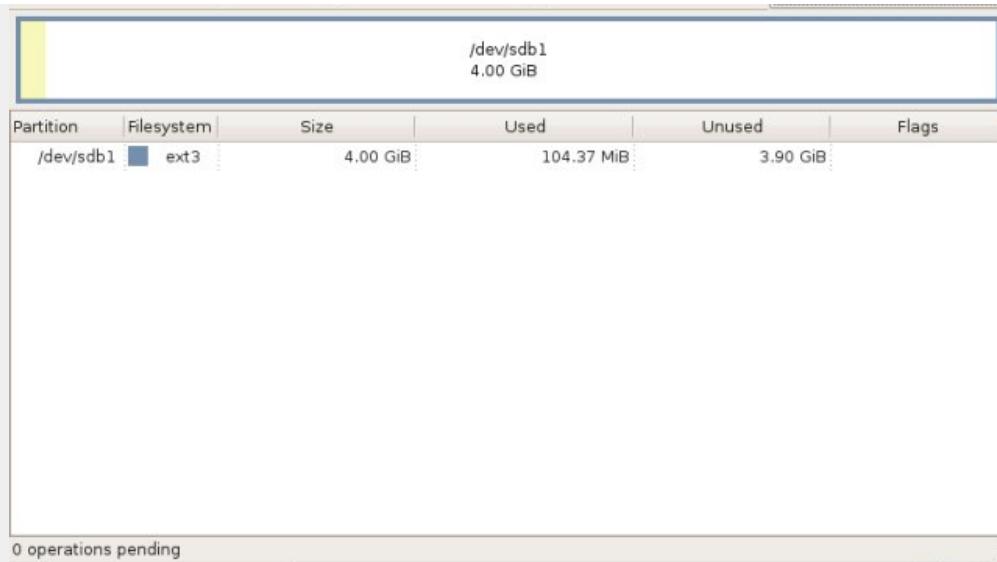
We have NTFS filesystem on the first partition (sda1). It's a primary partition. This is most likely a Windows C: drive. It also has the boot flag. We won't touch it.

The second in the list is sda2, the Extended partition, marked with lba flag as it is larger than 8GB. Inside it, we have three more NTFS partitions, which are likely D:, E: and F: drives in Windows. These are logical partitions, therefore they start with number 5. Please note sda7 is NOT the seventh partition; it's fourth on the color bar! We won't be touching those either.

The last bit of unallocated (gray) space is used by the Windows system. Ignore it. It will always be there on systems with Windows. So, it's the second disk we want, **sdb**.

Second disk:





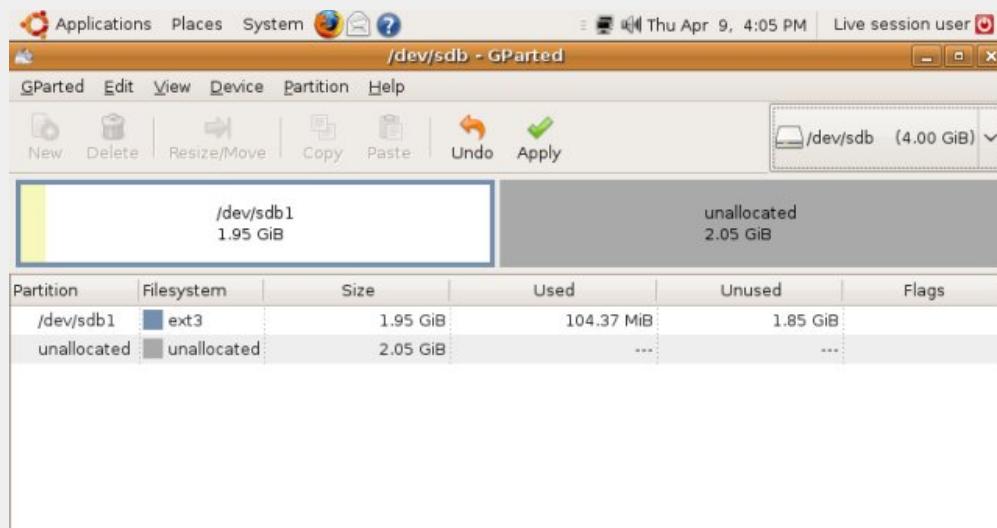
Currently, it has a single ext3 partition. It's most likely a left-over from an older installation or some testing. The partition is almost entirely empty, which makes it ideal for our games.

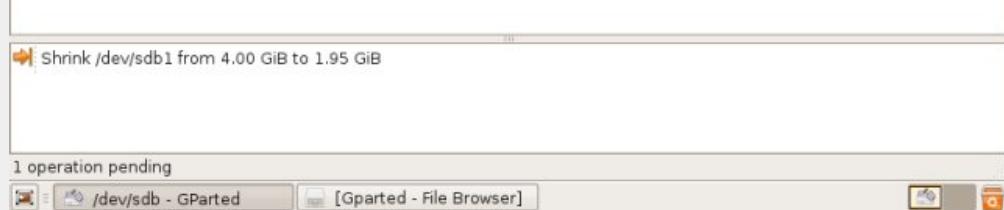
Task 1: Resize partition

This is the first thing we'll do. We'll shrink sdb1 to make space for more partitions. Highlight the partition and click on **Resize/Move** or in the menu, **Partition > Resize/Move**. Choose the new size. You can type in the numbers or drag the color bar.



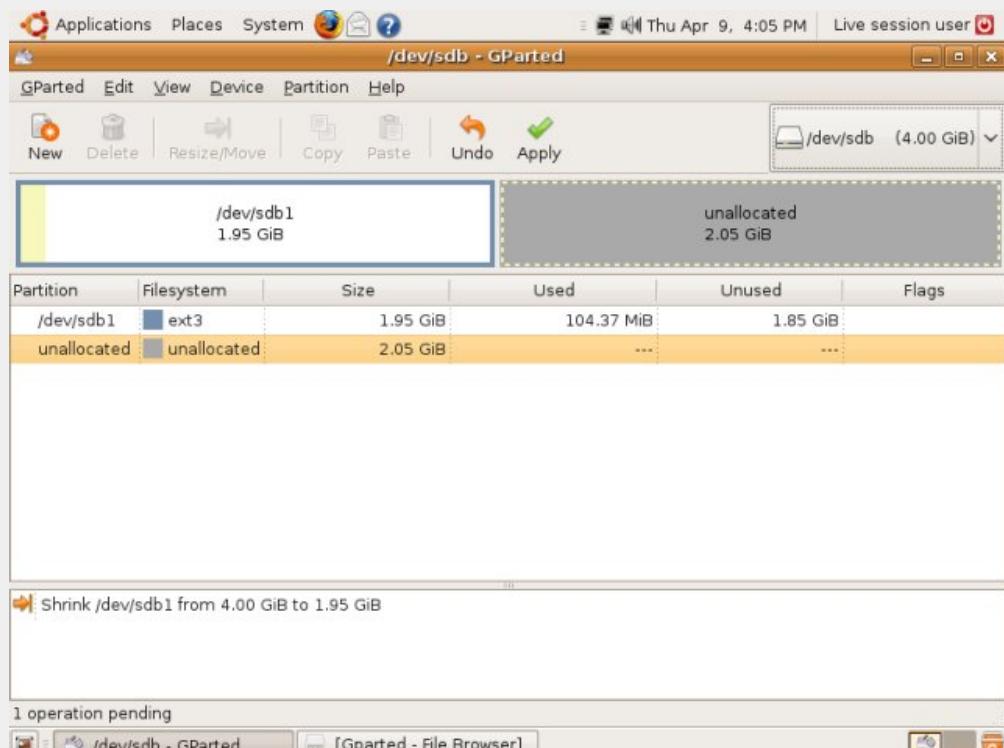
When the task is done, we will have freed approx. 2GB of space:



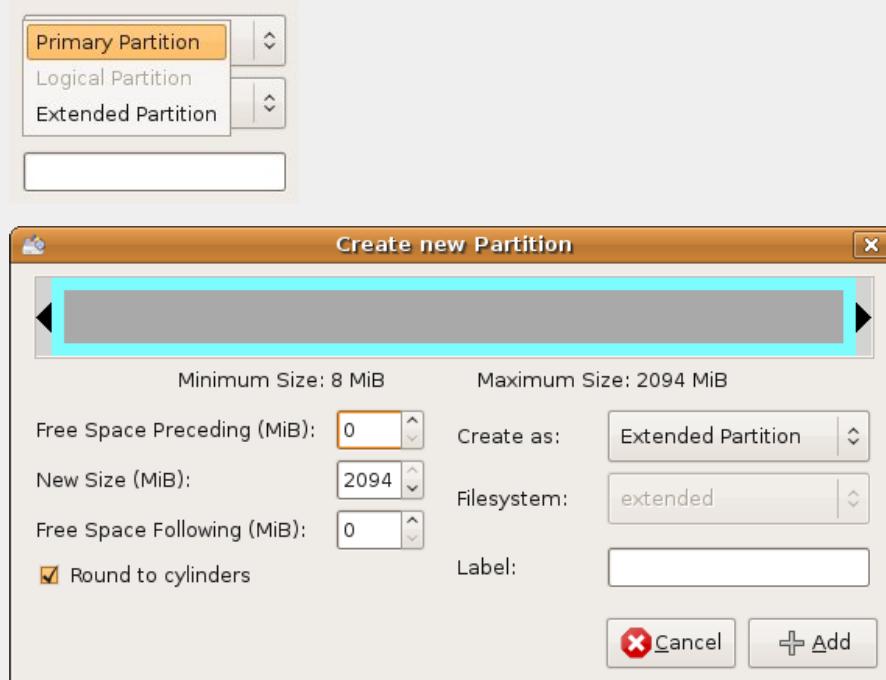


Task 2: Create new partition

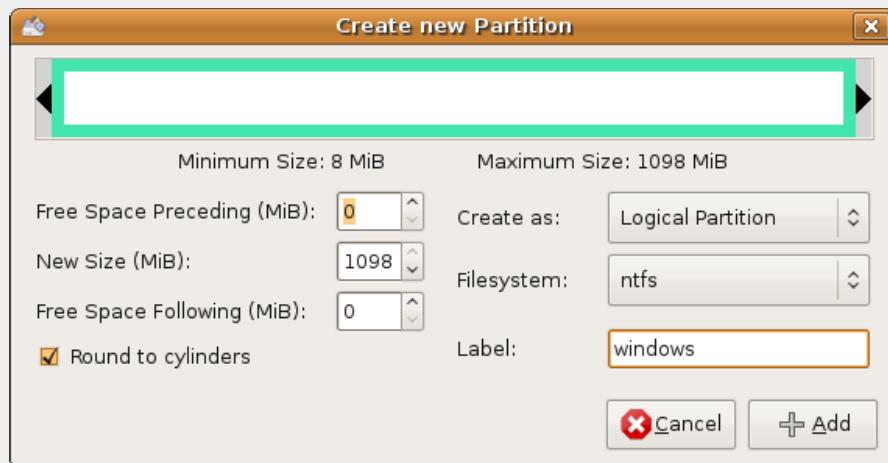
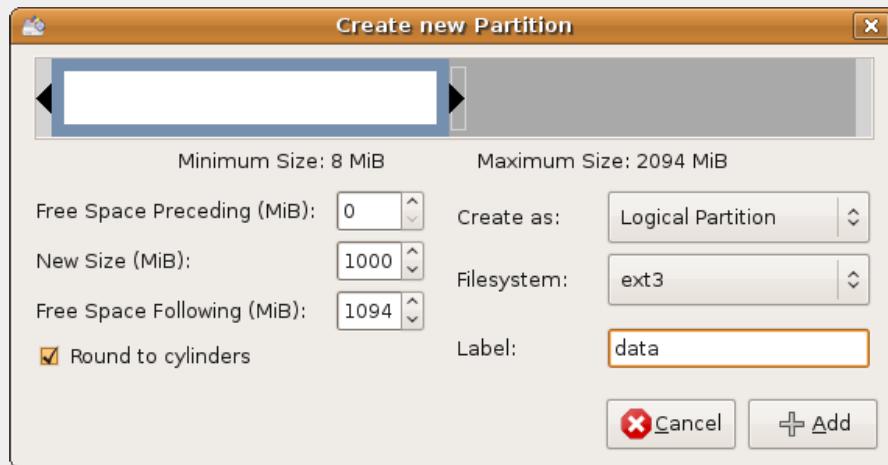
Now, we will create a new partition in the free, unallocated space after resized sdb1. We'll mark the free space and click on New.



In order not to waste the precious few primary partitions we have, we will create the Extended partition and then place other partitions inside it.



Then, we will create an Ext3 partition and an NTFS partition:



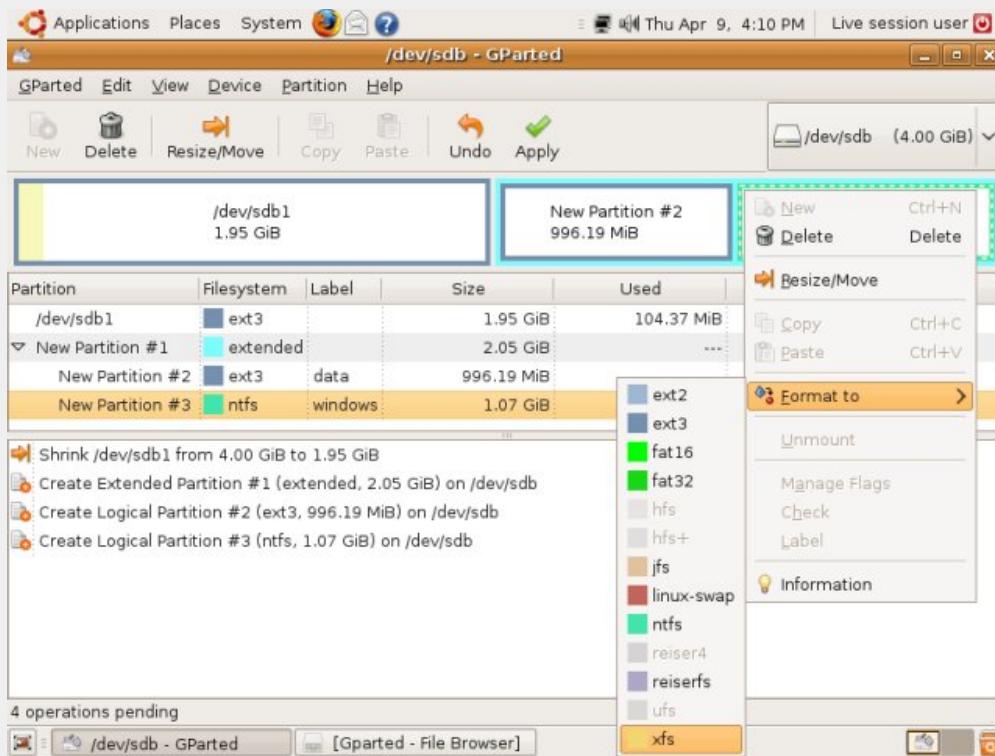
Please note I also added Labels to the two newly created partitions, so we can more easily identify them later. Here's our task list:

The screenshot shows the GParted main window for disk /dev/sdb. The disk is 4.00 GiB. It contains three partitions: /dev/sdb1 (1.95 GiB, ext3, data label), New Partition #1 (extended, 2.05 GiB), and New Partition #2 (logical, 996.19 MiB, ext3, data label). A fourth partition, New Partition #3 (logical, 1.07 GiB, ntfs, windows label), is currently selected. The 'Tasks' pane lists four pending operations: shrinking /dev/sdb1, creating an extended partition, creating a logical partition #2, and creating a logical partition #3. The status bar at the bottom indicates '4 operations pending'.

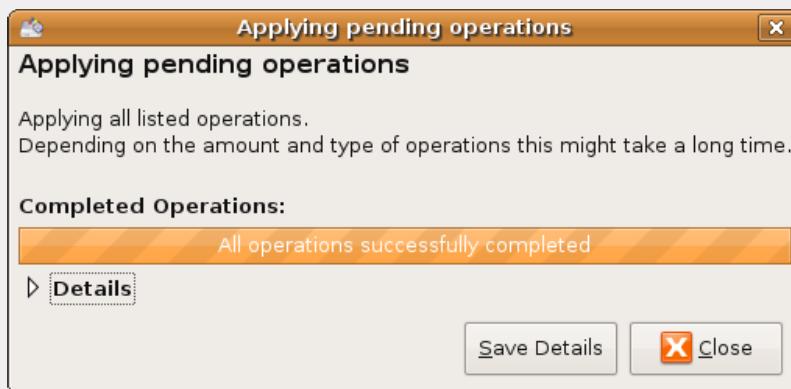
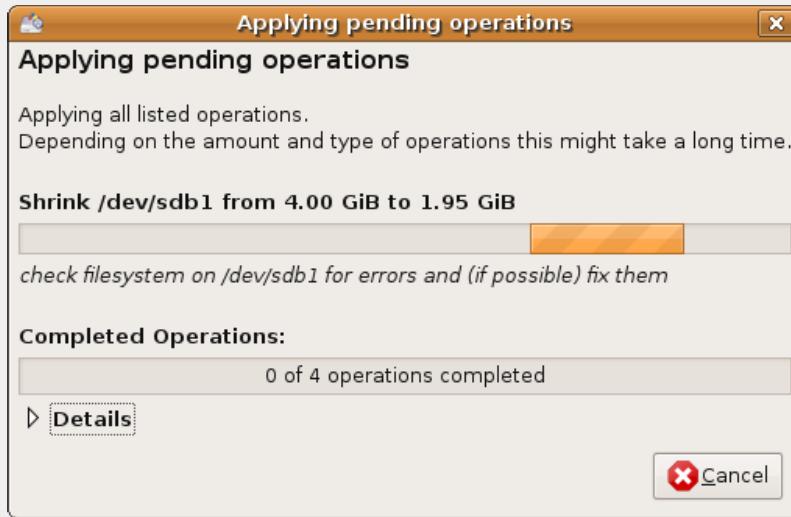
Please note that none of these tasks have taken place yet. Until you click **Apply**, none of the changes will be committed to the disk. This allows you to play freely. You will have the chance to confirm the

changes.

If you want to change the filesystem chosen for any which partition, you can do it without deleting the partition and creating a new one instead. You can simply format it with the new filesystem you desire. Either via the menu or by right-clicking on the partition, choose **Format to**. Notice the color legend. Each filesystem has a different color, making it more difficult to get confused.

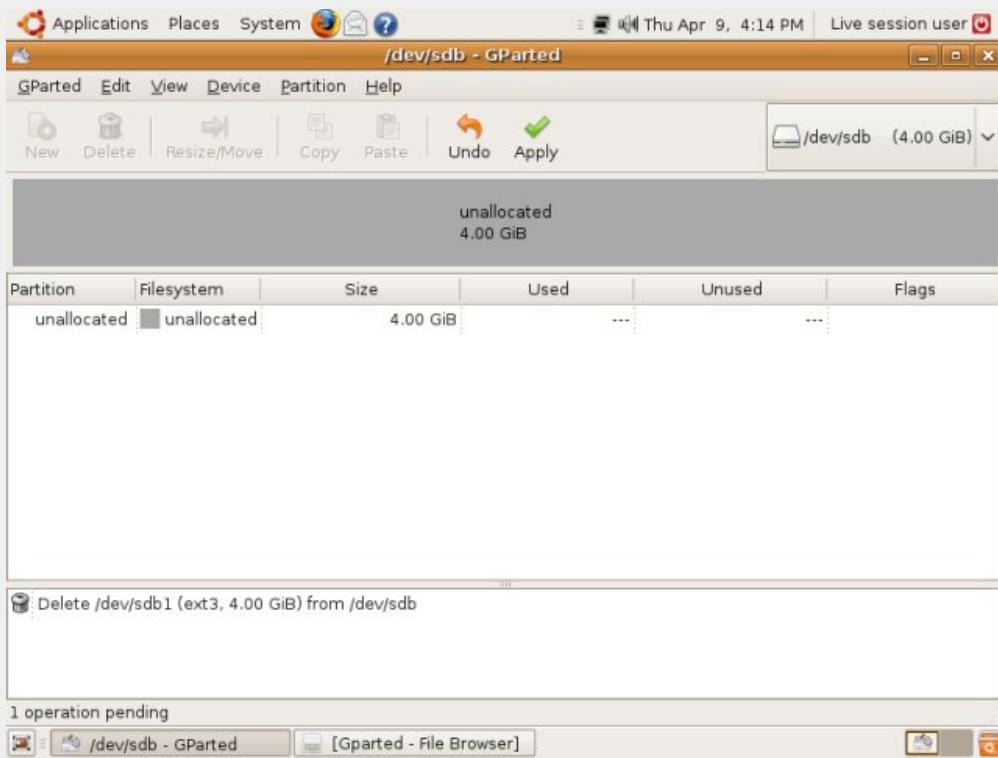


Once you click Apply, GParted will commit the changes:



Task 3: Delete partition

Sometimes, in order to grow or move partitions or create an alternative layout, you will have to delete partitions. Again, it's a very simple thing. Simply select the partition and click on **Delete**. It will be gone - still, again, you need to click on **Apply** to commit the changes. And you can also always **Undo** the operation.



Task 4: Create Partition Table

Empty hard disks will have no partition table - no "master" map defining the partitioning layout. Similarly, if you want to wipe the entire drive of existing partitions without manually deleting each one, you can simply reinitialize (recreate) the partition table. This is a drastic operation, so be careful when you do it:



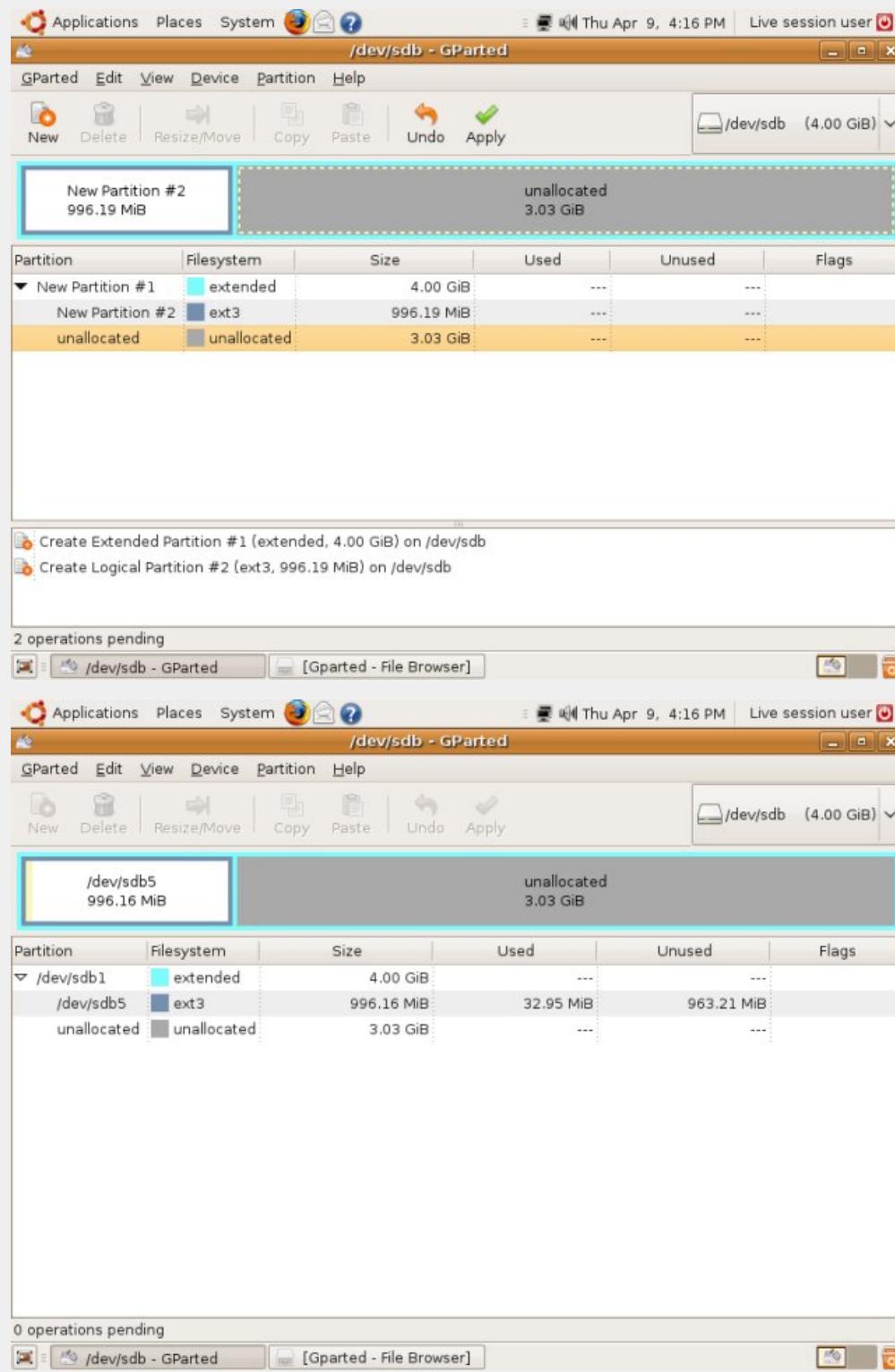
You will be warned:



Task 5: Create only Extended partition

This is an unusual setup, but it could happen. Your first partition won't be a primary partition used by

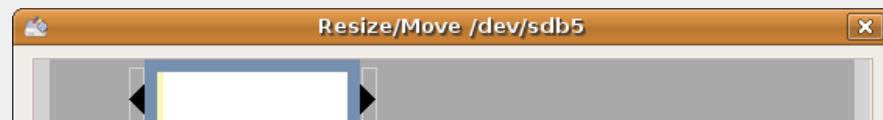
this or that operating system, it will be the Extended partition itself. The concept is the same as before:



Please note that sdb5 will be the first partition on the disk here!

Task 6: Move partition

You may also want to move partitions. This is not the most common task either, but you might need it. It's just like resizing, except that you specify the value for **Free Space Preceding** in the options.



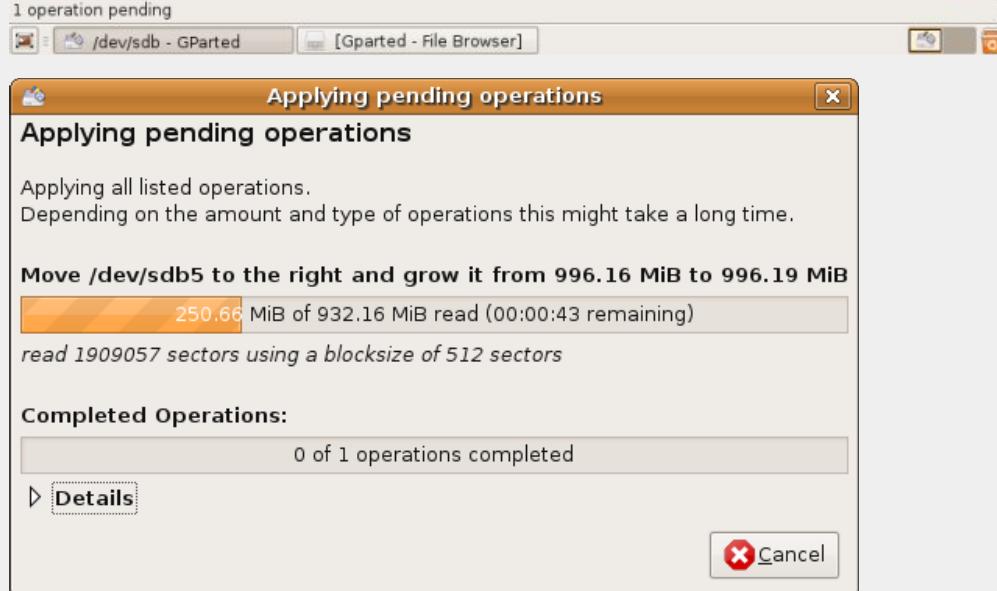
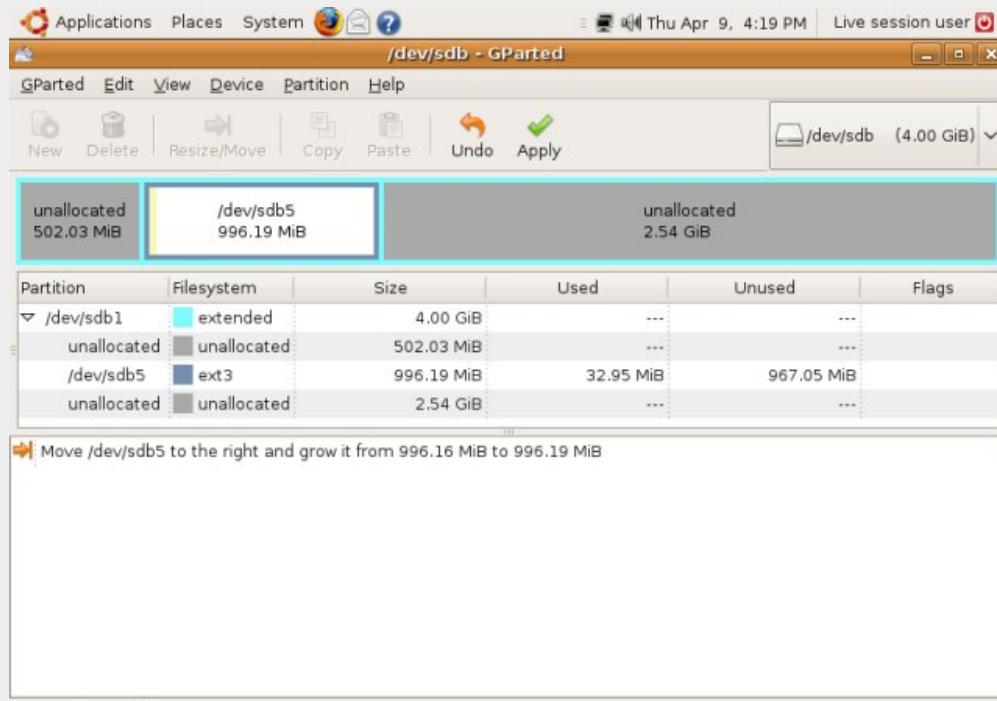
Minimum Size: 49 MiB Maximum Size: 4095 MiB

Free Space Preceding (MiB):

New Size (MiB):

Free Space Following (MiB):

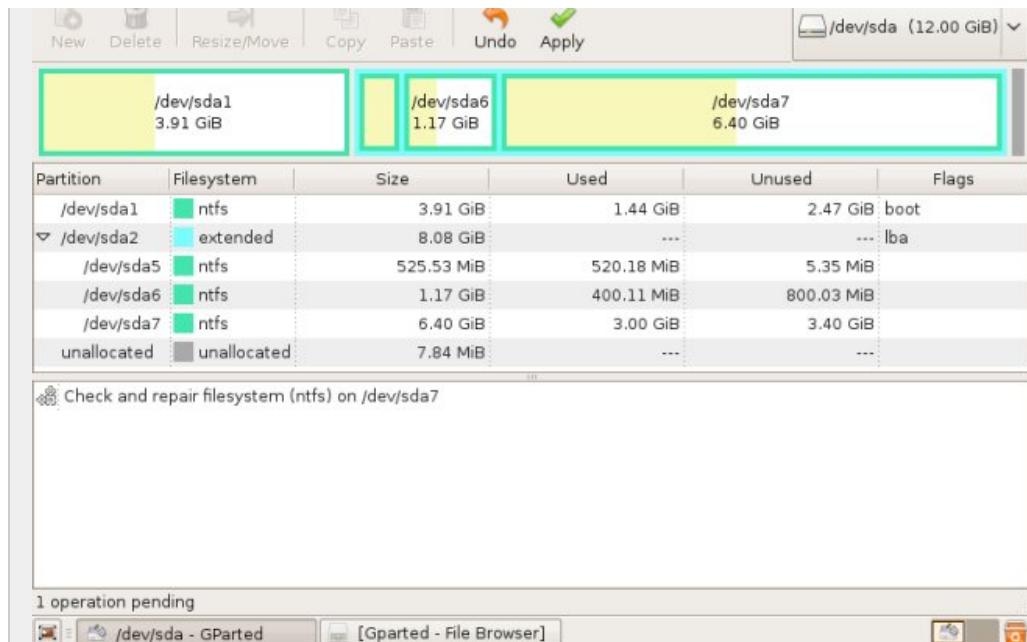
Round to cylinders



Task 7: Check & repair filesystem

GParted can also be used to try to fix errors on corrupt filesystems, like after a sudden power outage, for instance. Choose the relevant partition, right-click > **Check**.





Flags

Setting flags should usually be left to operating systems you're about to install, but you can do it yourself, if you want. Here's the list of all the flags GParted supports:



GParted capabilities

Wonder what filesystems can GParted work with? It gives you a nice graphical overview of its abilities. As you can see, it can do quite a lot with a large number of filesystems. Most notably, it works well with both FAT32 and NTFS, which is very important for Windows users.



Advanced tasks

This section is not strictly related to GParted. It's more of a bonus appendix, showing you a number of useful tricks that can enhance your partitioning skills. Here, though, we will have to leave the GUI behind and work with command line tools.

Change the Inode size

Inodes are data structure units that regulate how the filesystem will treat directories and files residing on it. A filesystem with small inodes will be able to house a very large number of files, but it won't have the best read/write performance. A filesystem with large inodes will be more suited for I/O throughput, but it won't be able to store too many files on it. Whatever the need, changing inodes cannot be done through the GParted GUI.

Why should you care?

That's a good question. Why would anyone be interested in changing the defaults set by the filesystem. Well, it turns out that some [imaging](#) software, like [Acronis True Image](#), can only work with Linux filesystems that use inodes of the 128-byte size. However, some modern distributions, like [Ubuntu 8.10 Intrepid Ibex](#), use 256-byte inodes, thus making the software unusable with this Ubuntu release.

This has caused quite a stir among the Acronis True Image users who happen to dual boot Windows and Linux and like to use their product to create system backups of both their operating systems.

The solution to the problem is very simple. First, we need to check what our filesystem currently uses. This is done using the [tune2fs](#) system utility.

```
(sudo) tune2fs -l /dev/<device-name> | grep "Inode size"
```

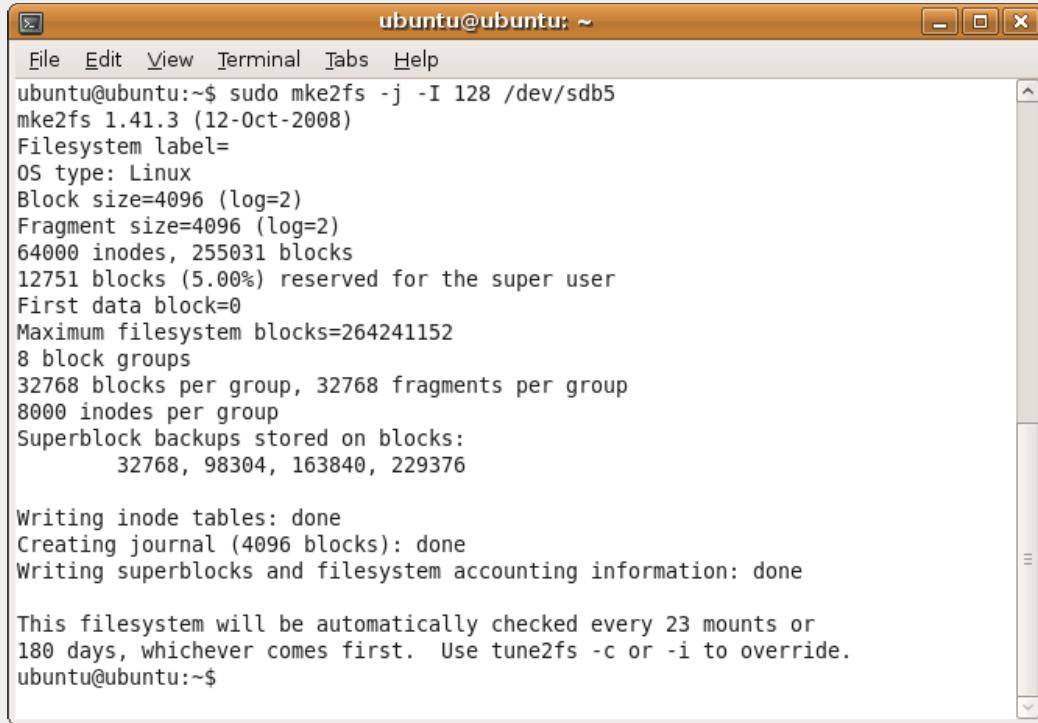
The above command polls the filesystems on the relevant /dev/ device for information. The grep command merely extracts the specific bit we need. Let's see what we get on our Ext3 filesystem formatted by GParted (our sdb5 from earlier):

```
ubuntu@ubuntu:~$ sudo tune2fs -l /dev/sdb5 | grep "Inode size"
Inode size: 256
ubuntu@ubuntu:~$
```

We have the Inode size: 256. Not good. We won't be able to use Acronis. So we need to change the size. This can be done using the **mke2fs** formatting utility for Ext2-based filesystems.

```
(sudo) mke2fs -j -I 128 /dev/sdb5
```

This will format the sd5 device as Ext3 filesystem (-j flag) with Inode size 128 (-I flag).

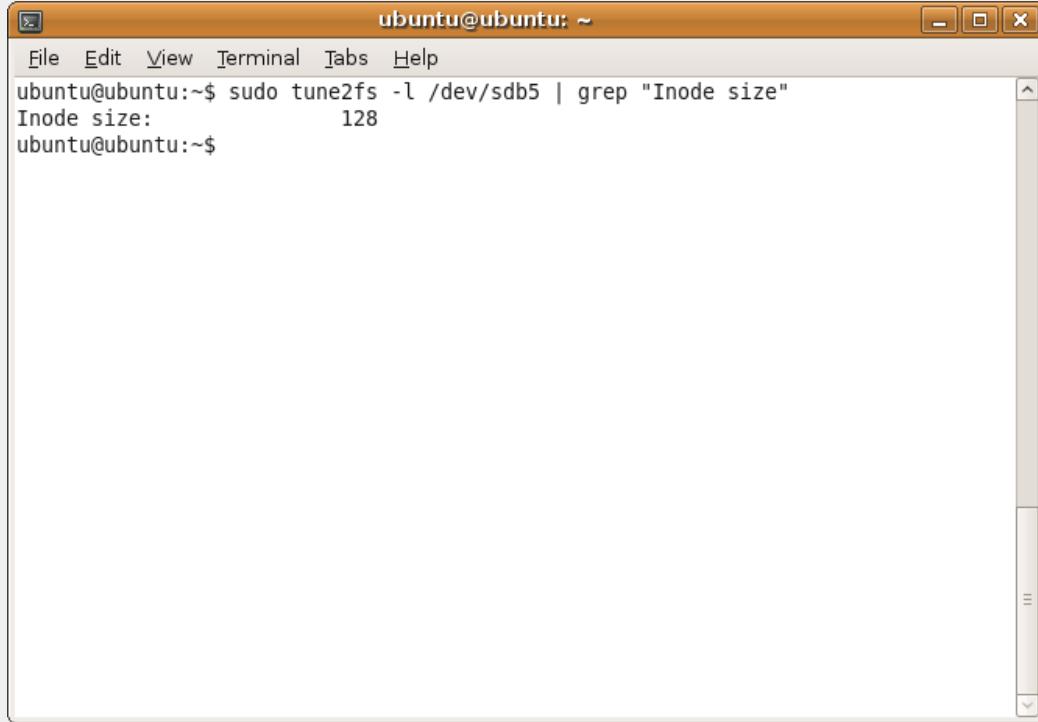


```
ubuntu@ubuntu: ~
File Edit View Terminal Tabs Help
ubuntu@ubuntu:~$ sudo mke2fs -j -I 128 /dev/sdb5
mke2fs 1.41.3 (12-Oct-2008)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
64000 inodes, 255031 blocks
12751 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=264241152
8 block groups
32768 blocks per group, 32768 fragments per group
8000 inodes per group
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
ubuntu@ubuntu:~$
```

Indeed, if we check again:



```
ubuntu@ubuntu: ~
File Edit View Terminal Tabs Help
ubuntu@ubuntu:~$ sudo tune2fs -l /dev/sdb5 | grep "Inode size"
Inode size:          128
ubuntu@ubuntu:~$
```

Our Inode size is good now. In general, I recommend all dual-boot users, especially those fond of imaging, to perform these steps manually on all partitions they intend to use for Linux and image.

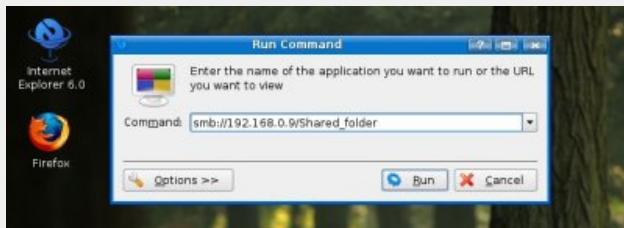
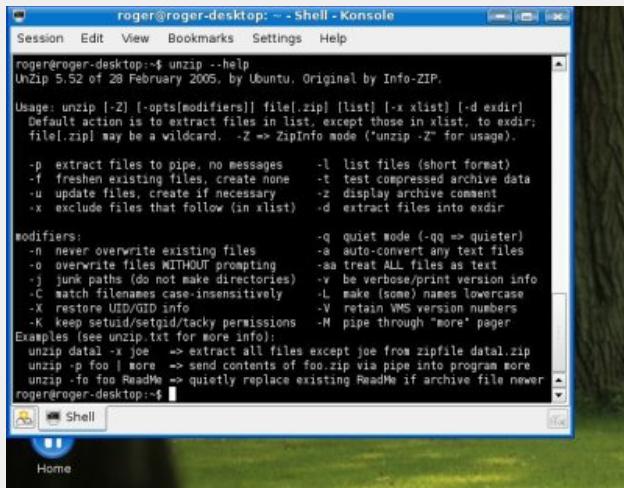
imaging, to perform these steps manually on all partitions they intend to use for Linux and image from Windows and/or using a Windows-based program like Acronis. That's about it. We now know the ins and outs of partitioning and working with GParted. Congratulations! Now, for some extras.

Recommended reading material

I most strongly recommend you at least take a look at the following articles. They are very detailed and thorough and should give you important information regarding the Linux operating system.

[Highly useful Linux commands & configurations](#)

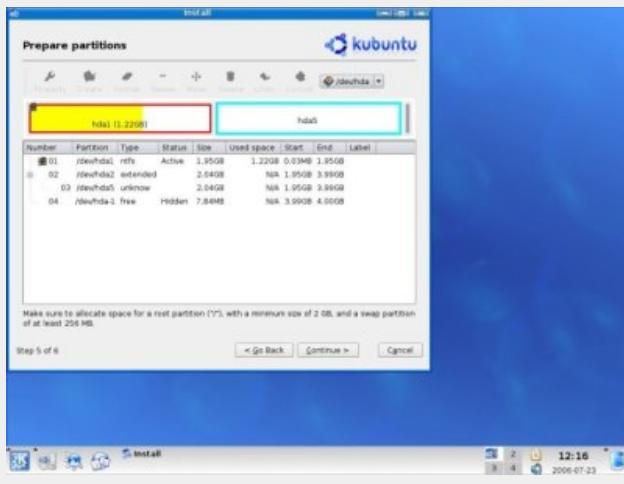
This tutorial will teach you the basic of Linux notation, command line usage, compilation, and setup of most common system configurations, like network and graphic drivers, printers, sharing, and more.



[Dual booting - Windows & Linux](#)

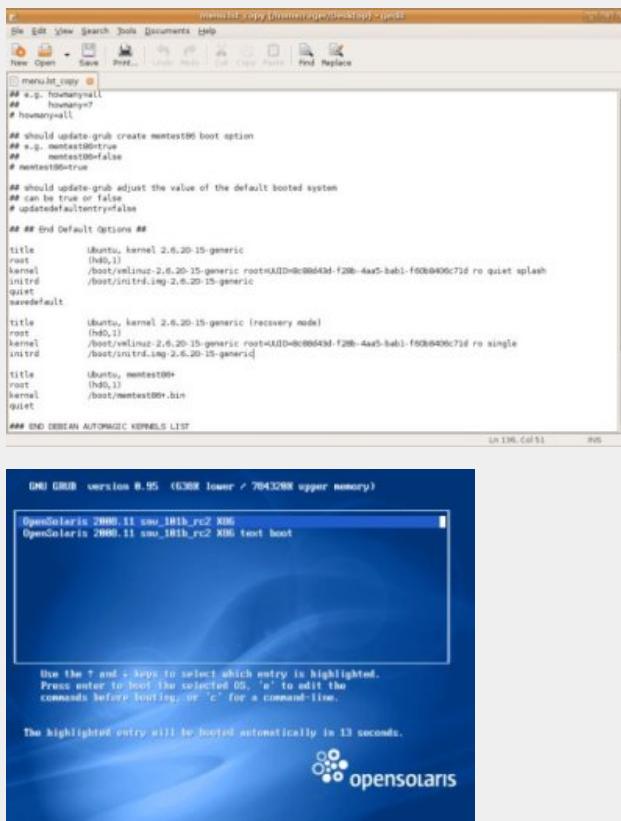
This tutorial demonstrates a side-by-side installation of [Windows XP](#) and [Kubuntu](#), a KDE flavor of the popular Ubuntu distro. Although the tutorial uses Kubuntu 6.06 as the demonstration platform, very little has changed in the releases since, at least when it comes to partitioning, especially the basic principles of it.

The two tutorials for Windows and Kubuntu, respectively, also detail the installation of these individual operating systems, so if you're not familiar with how this should be done, you're most welcome to read them.



[GRUB bootloader - Full tutorial](#)

This is a must-read article for anyone considering Linux or dual-booting with Windows. The tutorial explains the basic and advanced concepts of the bootloading procedure and tackles the most common issues arising from handling different operating systems and partitioning.



[Linux forensics - Introduction](#)

This article introduces the concepts of system rescue, recovery and investigation. Most importantly, it covers a number of critical tools for system administration than just about any computer user should be aware of. These include data and partition recovery utilities, GRUB restore utilities, backup and imaging software, and more.



[A \(cool\) list of Linux tools](#)

This article is a thorough compilation of a large number of great Linux software.

Conclusion

I hope this article will truly help you master the world of Linux, including the tricky, critical phase of working with partitions, especially during installations.

We've covered quite a bit, from creating of new partitions, to resizing, moving, deletion, labeling of partitions, we worked with different filesystems, including Ext3 and NTFS, we even dabbled some in

advanced command-line stuff like changing of inode size.

I do realize I have not covered every possible aspect of partitioning available, therefore, if you have suggestions or questions, feel free to email me; I will study your scenarios and possibly even update the tutorial to cover even more topics. I hope you liked it.

P.S. If you find this article useful, please [support](#) Dedoimedo.

Have fun!



[del.icio.us](#)

[stumble](#)

[digg](#)

[digg](#)

[reddit](#)

[slashdot](#)

[Top](#) [Home](#) [Terms of use](#) [Contact me](#) [About](#)

Copyright @ Dedoimedo.com 2006-2016; all rights reserved

[W3C](#) HTML 4.01

[W3C](#) CSS 2.0

[RSS](#) VALID FEEDS