# Smart Home

**Introduction**

Smart home technology, also known as [home automation](http://internetofthingsagenda.techtarget.com/definition/home-automation), provides homeowners security, comfort, convenience and energy efficiency by allowing them to control smart devices, often by a smart home app on their smartphone or other networked device. A part of the [internet of things (IoT)](http://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT), smart home systems and devices often operate together, sharing consumer usage data among themselves and automating actions based on the homeowners' preferences.

## 111Smart 1 Smart Home

### 1.1 Definition

A smart home is a residence that uses internet-connected devices to enable the remote monitoring and management of appliances and systems, such as lighting and heating…

### 1.2 Origins

The first smart homes were ideas, not actual structures. For decades, science fiction has explored the idea of smart home. Writers imagined a future where homes were interactive, and seemingly ran themselves. They described a smart home that continues to function even after humans have died out. It’s all well and frightening, until you consider the actual benefits of smart home, and then the idea becomes more comforting than chilling.

Although the idea of smart home has been around for some time, actual smart homes have only existed a short while. This timeline focuses on hardware, meaning actual inventions leading up to the smart homes we know today and can expect from the near future.

**1901 – 1920 –** **The invention of home appliances –** Although home appliances aren’t what we’d consider “smart,” they were an incredible achievement in the early twentieth century. These achievements began with the first engine-powered vacuum cleaner in 1901. A more practical electricity-powered vacuum was invented in 1907. Throughout two decades refrigerators would be invented, as well as clothes dryers, washing machines, irons, toasters, and so much more. It was a fantastic time for anyone who was employed as a maid by a very affluent family.  
  
**1966 - 1967 – ECHO IV and the Kitchen Computer** –Although it was never commercially sold, the ECHO IV was the first smart device. This clever device could compute shopping lists, control the home’s temperature and turn appliances on and off. The Kitchen Computer, developed a year later, could store recipes.

**1975-** Release of X10, a communication protocol for home automation, the smart home, once a pipe dream a la *The Jetsons*, came to life. X10 sends 120 kHz [radio frequency (RF)](http://searchnetworking.techtarget.com/definition/radio-frequency) bursts of digital information onto a home's existing electric wiring to programmable outlets or switches. These signals convey commands to corresponding devices, controlling how and when the devices operate. A transmitter could, for example, send a signal along the house's electric wiring, telling a device to turn on at a specific time.

**1998 – Early 2000s – Smart Homes** :Smart homes, or home automation, began to increase in popularity in the early 2000s. As such, different technology began to emerge. Smart homes suddenly became a more affordable option, and therefore a viable technology for consumers. Domestic technologies, home networking, and other gadgets began to appear on store shelves.

**Today’s Smart Homes**: Today’s smart homes are more about security and living greener. Our smart homes are sustainable, and they help to ensure that our homes aren’t expending unnecessary energy. They also help alert us to intruders (whether we’re at home or not).

Current trends in home automation include remote mobile control, automated lights, automated thermostat adjustment, scheduling appliances, mobile/email/text notifications, and remote video surveillance.  
“Connectivity and interactivity are driving the way families live and manage their homes. So, while we are expected to be in more places due to business travel, children’s school schedules and social activities, these new smart systems provide cutting edge connectivity to your household, even when you’re far away. And when the house is occupied, the high level of automation enables more convenience, control and safety from any part of your property. It all adds up to fewer worries and increased enjoyment of life, which is something we would all welcome.

### 1.3 Benefits

A lot of people are put off with the idea of a ‘Smart Home’ either because they think it will be too expensive, difficult to use or maybe they don’t feel like they will be in enough control of their own environment. However, Smart homes are becoming very affordable, due to having the option to build up little by little or have it all completed in one big project but with room for future improvement. Every device is at your control to make your life more convenient for your pleasure.

**1.Comfort**

One of the best parts of a Smart House is that you can set it up to be absolutely perfect for you. Have the lighting set up at just the right brightness, not too bright but not too dark either. You can even set it so that the lights slowly dims brighter instead of a sudden flash of blinding light you usually get or set up which lights/lamps turn on in the room depending on your activity.

**2. Security**

There are countless ways that a smart home could be keeping you secure at home. You can have motion sensors to detect movement after a determined time, such as at night or whilst you are at work. If the motion detectors are alerted you can be sent a notification to your smart phone, and/or have lights come on to make it seem as though someone is awake and scare the potential burglar away.

**3. Easily Controlled**

The most convenient part of Smart Homes is that every part of the house could now be at a touch of a screen. Whether that is your smart phone, tablet or a terminal built into your walls.

**4. Energy efficient**

The bonus of having everything in your house connected together means that everything is also monitored. Usually, programmed into Smart Homes are ‘Energy Saving’ options to your devices; such as not having the lights at full brightness depending how much natural light is in the room or having the thermostat adjusts itself throughout the day depending on the weather and if anyone is in the house or not.

**5. Accessibility**

Smart Homes are becoming a huge help for people with varying disabilities. People who cannot see can have voice activated interface to control their TV’s, lights, heating, anything that can be connected to electricity. People with muscle degenerating illness or disabilities may struggle reaching light switches, blinds or fire alarms but can now control such devices from a phone or a tablet. If you know someone in a wheelchair then you can set up automatic doors using sensors.

**6. Resale**

Lastly, look to the future. The new generation of home owners are likely going to be very interested in house that can be controlled via phones that they have grown up with all their lives.

### 1.4 Models

### There are Six Evolving Business Models for the Smart Home:

For decades, the promise of the smart home has tantalized businesses looking to capitalize on the dynamics of [connected technologies](https://www.jabil.com/solutions/by-industry/consumer/connected-consumer-technologies/connected-home.html?utm_source=iot-for-all&utm_medium=contributed-article&utm_campaign=jabil-connected-home-solutions&utm_content=five-evolving-business-models-for-the-smart-home) in the home, predicting and managing the desires of the occupant entirely without human interaction. Now, with the ascension of artificial intelligence from dream to near-reality and the growth of the [Internet of Things (IoT)](https://www.iotforall.com/what-is-iot-simple-explanation/), we’re closer than ever to realizing the dream.

Yes, the smart home is here to a certain extent, but experts predict the truly interconnected smart home, where devices and appliances can talk to each other may still be a decade away. There are important issues such as [data security and](https://www.iotforall.com/whats-holding-back-smart-home-mass-adoption/)interoperability between devices to address.

The challenge until then for manufacturers and service providers is how to maximize the revenue potential from stand-alone, pointed appliances and devices using smart home technologies that provide added-value to homeowners. The market is there. In 2016, 80 million smart home devices were delivered worldwide, a 64 percent increase from 2015, according to [IHS Markit](https://www.cnbc.com/2017/01/04/why-2017-will-finally-be-the-year-of-the-smart-home-consumers-figure-it-out.html).

Smart companies today understand that traditional models of one-time sell through of products, or standalone products that cannot be monitored intelligently, provide little or no opportunity to create a relationship with consumers and establish brand loyalty. They also realize that in today’s market, their products can serve as revenue generators with the data they gather based upon a homeowner’s energy usage, security connectivity and/or home entertainment preferences.

A good business model for companies looking to capitalize on the smart home revolution should contain many, if not all, of the following elements:

## 1- Personalization is Still King

One thing that will remain constant in the future of [smart home technologies](https://www.jabil.com/solutions/by-industry/consumer/connected-consumer-technologies/connected-home.html?utm_source=iot-for-all&utm_medium=contributed-article&utm_campaign=jabil-connected-home-solutions&utm_content=five-evolving-business-models-for-the-smart-home) is that companies must be prepared to create a unique user experience through engaging content and social media. Using data accumulated by a product, more opportunities exist for companies to craft messaging, plus deliver product and service information that is specific to the user’s tastes, past purchasing habits and product usage. Companies can provide access to social network-powered communities that generate positive experience feedback and share helpful tips and product recommendations.

## 2- Programmatic Commerce Generates Passive Revenue

Even though smart home technology is in its infancy, one thing has become clear: consumers are more than willing to delegate routine and repeated purchases of product accessories to their machines. People lead busy lives and letting a dishwasher order more soap or a coffee maker order more beans saves time.

More impressively, the smart appliance can even complete the transaction without any human input by simply accessing account information. Although these types of behaviors will need some time to be fully adopted due to security concerns or pure discomfort, they can significantly add to the long list of conveniences offered by the smart home.

## 3- Sell an Experience and Continual Value, Not Just One-Off Products

Products are being commoditized daily. Over the past decade, the “X-as-a-Service” model or subscription model has changed the way people buy everything from the music they listen to, the movies and sporting events they watch at home and even the apps they use on mobile devices.

It now appears consumers are willing to apply that same model when it comes to making smart home purchases. Rather than simply making a purchase and owning a device until it becomes obsolete or breaks down, they can opt to sign a monthly service agreement that provides for product use and upgrades as smart technology continues to evolve.

This business model provides consumers with the connectivity and technology they are growing to expect and offers services to make family life more secure and comfortable, while generating a predictable, monthly revenue stream for businesses.

## 4- Freemium is a Business Model

In addition to the subscription business model, highly disruptive companies are transforming their product management approach, and finely parsing solutions such that the base offering might be free, and a cross-sell/up-sell motion with new offers each month are constantly delivered to the user. The home monitoring base station may be “free,” but services such as burglar detection, carbon monoxide detection, smoke detection, etc. can be added as new offers.

Net/net, product managers should shift from long, drawn out BIG products with lots of features, to a lighter “base” offering, parsing out features into “options” that can be up-sold later.

## 5- Offer Savings and Safety

According to [Simmons Research](https://www.simmonsresearch.com/2017/09/13/rise-of-smart-home-consumer/), the most sought-after smart appliances by consumers provide increased energy efficiency and greater home security. Today’s consumers want thermostats that not only regulate temperature in the home, but can also make energy purchases on their own when prices are lowest. They want security systems that can be monitored remotely and offer connected motion detectors, surveillance cameras, automated door locks and other security system components that create a safer home. You can’t put a price on peace of mind.

## 6- Opportunities in Interconnectivity

With any new technology, the growing pains of development can present obstacles, which savvy business people realize are opportunities. This is especially true around interconnectivity. Developing products that can drive technology to closer interoperability between appliances, through advances in hub devices or those that further the development of smart home homeowner recognition (voice, iris or face) can help businesses stand apart from their competitors.

Another issue that will probably offer unique opportunities is transaction integration where companies can create innovative payment experiences, particularly when purchases are initiated by multiple smart appliances and all of them can be managed in one place.

### 1.5 Pros and Cons

As technology continues to grow and expand at an alarmingly fast rate, we’re starting to see more and more smart home gadgets hit the market as well. From smart thermostats that learn your routine to energy saving smart plugs that turn your devices off, the types of technology we see emerge every day are fascinating and increasingly complex.

The real question most homeowners have before they purchase these products is whether the investment is worth the price in the long run. Will investing in a smart thermostat, a smart wall plugin or a smart light bulb really save money on energy bills over time? Are these products truly safer for your family and your home? Are there any other incentives for purchasing smart home products?

Luckily, we can highlight the pros and cons of smart home technology and how it will affect your bills, insurance and the safety of your family in the long run.

### ****1-The Pros****

* **Saving money.**While the upfront investment might be costly, some smart home technology will save you money in the long run with energy efficient capabilities. For example, smart thermostats will learn and adapt to your behavior and schedule and adjust the temperature in your home accordingly, which will reduce costs on your monthly energy bills. Additionally, these products may increase the resale value of your home.
* **Convenience.**  
  Many of the smart home products available today can take over simple, everyday tasks such as starting your coffee in the morning or making sure your curling iron or hair straightener gets turned off after each use.
* **Remote monitoring.**With products such as smart plugs or smart bulbs, you can turn your lights or other electronic devices on or off at your convenience from wherever you might be. Whether you’re at work 20 minutes away or on vacation 2,000 miles away, you never have to worry if you left an electronic product on in your home again.
* **Insurance incentives.**Some insurers will actually reward homeowners who update their homes with smart technology by reducing their premiums. Other insurance providers will subsidize the cost of these devices or pay to install them. It all depends on the provider and coverage that you currently have.

### ****2- The Cons****

* **Complex programming and setup.**Many smart home products can be time consuming to set up and difficult to learn how to use. If this is a drawback for you, you may want to consider continuing to perform the tasks in the traditional way.
* **Different products on different networks.**Another drawback with many of the smart home technology is that different devices don’t necessarily communicate with each other. You may have to download and control each product from different apps and networks.
* **More money up front.**Upfront investments for smart home technology products and installation can be pricey. The long-term savings may be worth the cost for some products and not others. It’s best to do plenty of research on this before you purchase.
* **Safety concerns.**This is an extremely important con to smart home technology. While you can lock your doors or windows from a distance, you also may be at risk to hacking. Making sure your smart home devices connect to a hard-wired internet rather than Wi-Fi helps reduce that risk, but it’s still an important factor to consider.

Updating your current home systems with smart home technology is a completely personal decision that should be carefully considered. As the technology stands right now, there are some products that are worth the investment and some that may need a little more time to develop.

If you are considering smart home technology for your home, make sure to research specific products to find the best fit for you. If you ever have questions about how this new technology may affect your insurance rates, don’t hesitate to [give us a call](http://lindowinsurance.com/contact/). We’re always happy to help protect you and your family!.

**CHAPTER 2**

**Method and Result**

**Introduction**

A Smart home is a dwelling containing interconnected components. The components could be things to control inside the house, for example lights or heating. They could also be things to get information from, for example the current temperature inside the house, or if windows and doors are opened or closed . A Smart home uses these components to improve the quality of life for its users. However, a high tech-equipped house does not have to be a Smart home, Smart homes with smartphones 2 Background tech-equipped residence by ”the network through which each of the technological components and information about them is connected and coordinated”. There are many definitions of Smart homes, Alam define the concept of Smart homes as an application of ubiquitous computing, with the purpose to provide user context-aware services. These services can be automated or assistive in the form of ambient intelligence, remote control, or home automation. Ambient intelligent services use sensors to monitor the user context, and then take actions to ease the user’s daily life. An example could be an automatic heating regulation that lowers the temperature when no one is occupying the house. Remote control services let users monitor and control devices inside the house. This could be done from within the house but also when the users are not at home. An example of remote control services could be to get information about which windows and doors are open, and even being able to close them remotely. Home automation services are used for when the users want a schematic control of the house. This could be timer planed events like having the lights on during specified time intervals, or logical planed events like having lights on whenever a specific sensor is triggered. Based on a Smart home service main purpose, the service can be categorized into three basic categories; security, energy consumption and management, and lifestyle support Most Smart home projects that have been conducted over the past decades can be categorized into one or more of these categories with respect to which user’s needs they targeted. Grouped Smart home services even further. the three broad yet interconnected groups; Safety, energy consumption and management, and Lifestyle support contains the Smart home service categories security, assisted living, healthcare, entertainment, communication, comfort and convenience, and energy efficiency.

**2.1 Design Process**

The design process was constructed towards the scope of this project. The process was based on methods from the design field researched by us and contains the following steps:

1. Competitive analysis

2. Setting usability goals

3. Participatory design

4. Coordinated design of the total interface Standards Product identity

5. Guidelines and heuristic analysis

6. Prototyping

7. Empirical testing

8. Iterative design Capture the design rationale

**2.2 Literature Study**

The project started with a pre-study where basic information about the Smart homes was collected. The goal with the pre-study was to get an overview of Smart home technology in general. The pre-study also eased the search for material to use as reference in this thesis, such as information about Smart home technology. The information came largely from conference articles, and scientific papers about related to Smart homes and home automation. The main point of this research was to acquire a broad knowledge base to use as a foundation during the ideation in a later stage of the project.

Since the prototype being created in this project was a Smart home application that would either use an ESP8266 related with the electronics devices in the house and an android application that allows us to manage our smart home.

**2.3 Smart Home Application**

When the literature study was completed, we decided on what type of Smart home application that would be developed. It was an active decision not to involve the users in this stage, due to that the users does not always know what they want. Namely that the user’s imagination limits the answers when the user is asked what he or she wants. We were interested in an application which could control light’s door’s and gas measure and temperature in the house using the ESP8266 microchip which is connected with the router. Therefore, it was chosen that the Smart home application would be a communication application. The application’s core features would be to control and display information about its environment. To read information about its environment, the application would connect to a ESP8266t, from which it could control switching on light and closing or opening door’s and collect data like temperature and if door is opened or closed and light’s is on or off. The data would then be shown in the application, Since the application would be used in a home, the displaying of data would have to be performed in a most calm way. This is because the application should not disturb the occupants of its environment. Instead, the application should display the data with smooth changes in the phones. This keeps the information away from the user’s center of attention when it is not needed. Beside the applications core feature, it would also need to provide its users with the ability to choose which other users to follow, and to allow to be followed by. Without this function, the users privacy would easily be intruded by unwanted users. The user-following functionality would allow users to choose by themselves what to do with the application.

**2.4 Android**

**2.4.1 Introduction**

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open source software and designed primarily for touchscreen mobile devices such as smartphones and tablets. In addition, Google has further developed Android TV for televisions, Android Auto for cars, and Wear OS for wrist watches, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics.

Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The operating system has since gone through multiple major releases, with the current version being 8.1 "Oreo", released in December 2017. The core Android source code is known as Android Open Source Project (AOSP), and is primarily licensed under the Apache License.

Android is also associated with a suite of proprietary software developed by Google, including core apps for services such as Gmail and Google Search, as well as the application store and digital distribution platform Google Play, and associated development platform. These apps are licensed by manufacturers of Android devices certified under standards imposed by Google, but AOSP has been used as the basis of competing Android ecosystems, such as Amazon.com's Fire OS, which utilize its own equivalents to these Google Mobile Services.

Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2017, it has over two billion monthly active users, the largest installed base of any operating system, and as of 2017, the Google Play store features over 3.5 million apps.

**2.4.2 Development**

Android is developed by Google until the latest changes and updates are ready to be released, at which point the source code is made available to the Android Open Source Project (AOSP),[123] an open source initiative led by Google.[124] The AOSP code can be found without modification on select devices, mainly the Nexus and Pixel series of devices.[125] The source code is, in turn, customized and adapted by original equipment manufacturers (OEMs) to run on their hardware.[126][127] Also, Android's source code does not contain the often proprietary device drivers that are needed for certain hardware components.[128] As a result, most Android devices, including Google's own, ultimately ship with a combination of free and open source and proprietary software, with the software required for accessing Google services falling into the latter category.

**2.4.3 Software stack**

On top of the Linux kernel, there are the middleware, libraries and APIs written in C, and application software running on an application framework which includes Java-compatible libraries. Development of the Linux kernel continues independently of Android's other source code projects.

Until version 5.0, Android used Dalvik as a process virtual machine with trace-based just-in-time (JIT) compilation to run Dalvik "dex-code" (Dalvik Executable), which is usually translated from the Java bytecode. Following the trace-based JIT principle, in addition to interpreting the majority of application code, Dalvik performs the compilation and native execution of select frequently executed code segments ("traces") each time an application is launched.[177][178][179] Android 4.4 introduced Android Runtime (ART) as a new runtime environment, which uses ahead-of-time (AOT) compilation to entirely compile the application bytecode into machine code upon the installation of an application. In Android 4.4, ART was an experimental feature and not enabled by default; it became the only runtime option in the next major version of Android, 5.0.[180]

For its Java library, the Android platform uses a subset of the now discontinued Apache Harmony project.[181] In December 2015, Google announced that the next version of Android would switch to a Java implementation based on the OpenJDK project.[182]

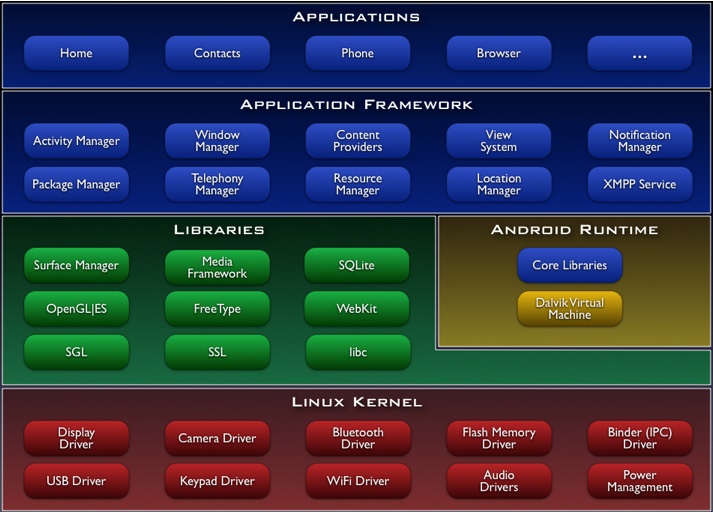
Android's standard C library, Bionic, was developed by Google specifically for Android, as a derivation of the BSD's standard C library code. Bionic itself has been designed with several major features specific to the Linux kernel. The main benefits of using Bionic instead of the GNU C Library (glibc) or uClibc are its smaller runtime footprint, and optimization for low-frequency CPUs. At the same time, Bionic is licensed under the terms of the BSD licence, which Google finds more suitable for the Android's overall licensing model.[179]

Aiming for a different licensing model, toward the end of 2012, Google switched the Bluetooth stack in Android from the GPL-licensed BlueZ to the Apache-licensed BlueDroid.[183]

Android does not have a native X Window System by default, nor does it support the full set of standard GNU libraries. This made it difficult to port existing Linux applications or libraries to Android,[174] until version r5 of the Android Native Development Kit brought support for applications written completely in C or C++.[184] Libraries written in C may also be used in applications by injection of a small shim and usage of the JNI.[185

Since Marshmallow, "Toybox", a collection of command line utilities (mostly for use by apps, as Android doesn't provide a command line interface by default), replaced similar "Toolbox" collection.

Android has another operating system, Trusty OS, within it, as a part of "Trusty" "software components supporting a Trusted Execution Environment (TEE) on mobile devices." "Trusty and the Trusty API are subject to change. [..] Applications for the Trusty OS can be written in C/C++ (C++ support is limited), and they have access to a small C library. [..] All Trusty applications are single-threaded; multithreading in Trusty userspace currently is unsupported. [..] Third-party application development is not supported in" the current version, and software running on the OS and processor for it, run the "DRM framework for protected content. [..] There are many other uses for a TEE such as mobile payments, secure banking, full-disk encryption, multi-factor authentication, device reset protection, replay-protected persistent storage, wireless display ("cast") of protected content, secure PIN and fingerprint processing, and even malware detection.



**2.4.4 Platform usage**

Charts in this section provide breakdowns of Android versions, based on devices accessing the Google Play Store in a seven-day period ending on May 7, 2018.[357][b] Therefore, these statistics exclude devices running various Android forks that do not access the Google Play Store, such as Amazon's Fire tablets.

.

| **Version** | **Code name** | **Release date** | **API level** | **ART/DVM** | **Distribution** | **First devices to run version** |
| --- | --- | --- | --- | --- | --- | --- |
| [**8.1**](https://en.wikipedia.org/wiki/Android_version_history#Android_8.1_Oreo_(API_27)) | [Oreo](https://en.wikipedia.org/wiki/Android_Oreo) | December 5, 2017 | **27** | ART | 0.8% | [Pixel,](https://en.wikipedia.org/wiki/Pixel_(smartphone)) [Pixel XL](https://en.wikipedia.org/wiki/Pixel_XL), [Nexus 6P](https://en.wikipedia.org/wiki/Nexus_6P), [Nexus 5X](https://en.wikipedia.org/wiki/Nexus_5X) |
| [**8.0**](https://en.wikipedia.org/wiki/Android_version_history#Android_8.0_Oreo_(API_26)) | August 21, 2017 | **26** | [ART](https://en.wikipedia.org/wiki/Android_Runtime) | 4.9% | N/A |
| [**7.1**](https://en.wikipedia.org/wiki/Android_version_history#Android_7.1_Nougat_(API_25)) | [Nougat](https://en.wikipedia.org/wiki/Android_Nougat) | October 4, 2016 | 25 | [ART](https://en.wikipedia.org/wiki/Android_Runtime) | 8.2% | [Pixel](https://en.wikipedia.org/wiki/Pixel_(smartphone)), [Pixel XL](https://en.wikipedia.org/wiki/Pixel_XL) |
| [**7.0**](https://en.wikipedia.org/wiki/Android_version_history#Android_7.0_Nougat_(API_24)) | August 22, 2016 | 24 | ART | 22.9% | [Nexus 5X](https://en.wikipedia.org/wiki/Nexus_5X), [Nexus 6P](https://en.wikipedia.org/wiki/Nexus_6P) |
| [**6.0**](https://en.wikipedia.org/wiki/Android_version_history#Android_6.0_Marshmallow_(API_23)) | [Marshmallow](https://en.wikipedia.org/wiki/Android_Marshmallow) | October 5, 2015 | 23 | ART | 25.5% |
| [**5.1**](https://en.wikipedia.org/wiki/Android_version_history#Android_5.1_Lollipop_(API_22)) | [Lollipop](https://en.wikipedia.org/wiki/Android_Lollipop) | March 9, 2015 | 22 | ART | 17.6% | [Android One](https://en.wikipedia.org/wiki/Android_One) |
| [**5.0**](https://en.wikipedia.org/wiki/Android_version_history#Android_5.0_Lollipop_(API_21)) | November 3, 2014 | 21 | ART 2.1.0 | 4.8% | [Nexus 6](https://en.wikipedia.org/wiki/Nexus_6), [Nexus 9](https://en.wikipedia.org/wiki/Nexus_9) |
| [**4.4**](https://en.wikipedia.org/wiki/Android_version_history#Android_4.4_KitKat_(API_19)) | [KitKat](https://en.wikipedia.org/wiki/Android_KitKat) | October 31, 2013 | 19 | [DVM](https://en.wikipedia.org/wiki/Dalvik_(software)) (and ART 1.6.0) | 10.3% | [Nexus 5](https://en.wikipedia.org/wiki/Nexus_5) |
| [**4.3**](https://en.wikipedia.org/wiki/Android_version_history#Android_4.3_Jelly_Bean_(API_18)) | [Jelly Bean](https://en.wikipedia.org/wiki/Android_Jelly_Bean) | July 24, 2013 | 18 | DVM | 0.6% | [Nexus 7 2013](https://en.wikipedia.org/wiki/Nexus_7_(2013)) |
| [**4.2**](https://en.wikipedia.org/wiki/Android_version_history#Android_4.2_Jelly_Bean_(API_17)) | November 13, 2012 | 17 | DVM | 2.2% | [Nexus 4](https://en.wikipedia.org/wiki/Nexus_4), [Nexus 10](https://en.wikipedia.org/wiki/Nexus_10) |
| [**4.1**](https://en.wikipedia.org/wiki/Android_version_history#Android_4.1_Jelly_Bean_(API_16)) | July 9, 2012 | 16 | DVM | 1.5% | [Nexus 7](https://en.wikipedia.org/wiki/Nexus_7_(2012)) |
| [**4.0**](https://en.wikipedia.org/wiki/Android_version_history#Android_4.0_Ice_Cream_Sandwich_(API_14)) | [Ice Cream Sandwich](https://en.wikipedia.org/wiki/Android_Ice_Cream_Sandwich) | October 19, 2011 | 15 | DVM | 0.4% | [Galaxy Nexus](https://en.wikipedia.org/wiki/Galaxy_Nexus) |
| [**2.3**](https://en.wikipedia.org/wiki/Android_version_history#Android_2.3.3_Gingerbread_(API_10)) | [Gingerbread](https://en.wikipedia.org/wiki/Android_Gingerbread) | February 9, 2011 | 10 | DVM 1.4.0 | 0.3% | [Nexus S](https://en.wikipedia.org/wiki/Nexus_S) |

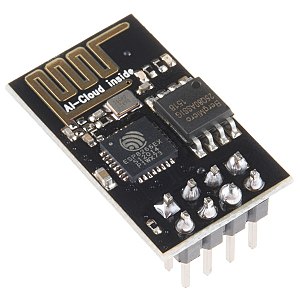
**2.5 ESP8266**

**2.5.1 Introduction**

The ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturer, Espressif Systems.

The chip first came to the attention of western makers in August 2014 with the ESP-01 module, made by a third-party manufacturer, Ai-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands. However, at the time there was almost no English-language documentation on the chip and the commands it accepted.[2] The very low price and the fact that there were very few external components on the module which suggested that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, chip, and the software on it, as well as to translate the Chinese documentation.

The ESP8285 is an ESP8266 with 1 MiB of built-in flash, allowing for single-chip devices capable of connecting to Wi-Fi.



**2.5.2 Features**

* + - Processor: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz†
    - Memory:
    - 32 KiB instruction RAM
    - 32 KiB instruction cache RAM
    - 80 KiB user data RAM
    - 16 KiB ETS system data RAM
    - External QSPI flash: up to 16 MiB is supported (512 KiB to 4 MiB typically included)
    - IEEE 802.11 b/g/n Wi-Fi
    - Integrated TR switch, balun, LNA, power amplifier and matching network
    - WEP or WPA/WPA2 authentication, or open networks
    - 16 GPIO pins
    - SPI
    - I²C (software implementation)[5]
    - I²S interfaces with DMA (sharing pins with GPIO)
    - UART on dedicated pins, plus a transmit-only UART can be enabled on GPIO2
    - 10-bit ADC (successive approximation ADC)

**2.5.3 PinOut of Version-01**

The Pinout is as follows for the 1st basic module,

1. VCC, Voltage (+ 3.3 V (upto 3.6 V it can handle))
2. GND, Ground (0 V)
3. RX, Receive data bit X
4. TX, Transmit data bit X
5. CH\_PD, Chip Power Down
6. RST, Reset
7. GPIO 0, General Purpose Input-Output No. 0
8. GPIO 2, General Purpose Input-Output No. 2

**2.5.4 SDKs**

In late October 2014, Espressif Systems released a software development kit (SDK) for programming the chip directly which removed the need for a separate microcontroller.[6] Since then, there have been many official SDK releases from Espressif; Espressif maintains two versions of the SDK – one that is based on FreeRTOS and the other based on callbacks.[7]

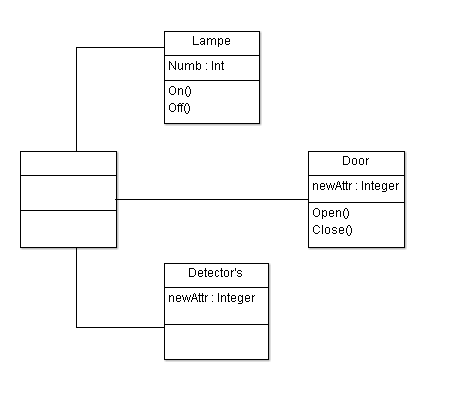
An alternative to Espressif's official SDK is the open source ESP-Open-SDK[8] that is based on the GCC toolchain. ESP8266 uses the Cadence Tensilica L106 microcontroller and the GCC toolchain is open-sourced and maintained by Max Filippov.[9] Another alternative is the "Unofficial Development Kit" by Mikhail Grigorev.[10][11]

Other SDKs (mostly open source) include:

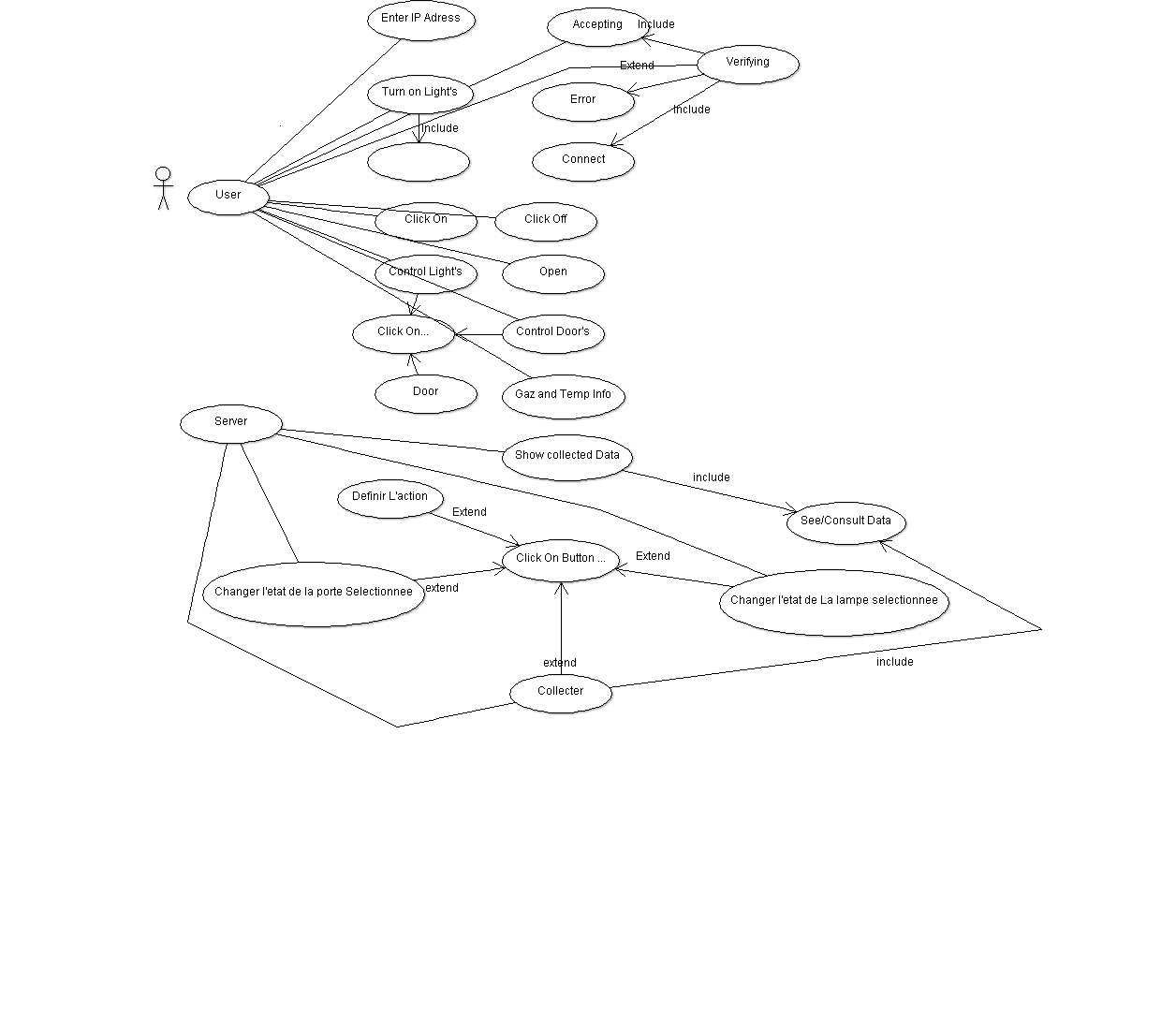
* + - NodeMCU – A Lua-based firmware.
    - Arduino – A C++ based firmware. With this core, the ESP8266 CPU and its Wi-Fi components can be programmed like any other Arduino device. The ESP8266 Arduino Core is available through GitHub.
    - Sming - An actively developed asynchronous C/C++ framework with superb performance and multiple network features.
    - PlatformIO (https://platformio.org/platforms/espressif8266) – A cross-platform IDE and unified debugger which sits on top of Arduino code and libraries.
    - MicroPython – A port of MicroPython (an implementation of Python for embedded devices) to the ESP8266 platform.
    - ESP8266 BASIC – An open source basic interpreter specifically tailored for the internet of things. Self hosting browser based development environment.
    - Zbasic for ESP8266 – A subset of Microsoft's widely used Visual Basic 6 which has been adapted as a control language for the ZX microcontroller family and the ESP8266.
    - Espruino – An actively maintained JavaScript SDK and firmware, closely emulating Node.js. Supports a few MCUs, including the ESP8266.
    - Mongoose OS – An open source Operating System for connected products. Supports ESP8266 and ESP32. Develop in C or JavaScript.[12]

**2.6 UML(Unified Modeling Language)**

**2.6.1 Diagram of Class**



**2.6.2 Diagram Of utilization**

****

**Project Realization**

**3.1 Objectives**

Starting from our problem, this project has a lot of objectives that we should be able to complete at the end of this project, so we can control electronics appliances in the house easily with our Smartphone’s.

**3.2 App Functionality**

The first time when you install the app you got a home screen picture which is telling you to wait while the app connect with server on the ESP8266 then you’ll find 3 fragment every one handle something so the first fragment control lights and show the temp and movement detection the second fragment control doors and the last one is for the remote control so you can control your TV , receiver …

**3.3 Realization of project**

**3.3.1 Tools**

ESP8266

Electrical cables

Movement detector

Lamps

4 Channel 5V Relay

Relay table

UML

Arduino IDE for linux

Android studio for Linux

Android SDK

Apache

My SQL

Photoshop

**3.3.2 Limitation**

Our Application it’s compatible with the android Smartphone who support API 15 to API 27 ( Oreo ), we’ve tested the application on two Smartphone Sony Experia Z3 and P6 LTE and with the Emulator .

**3.3.3 Android App**

**3.3.3.1 Welcome Screen**

To accomplish this project application and manipulates all the interfaces of our

Application we need two types of file {Java Classes, Xml Files}.

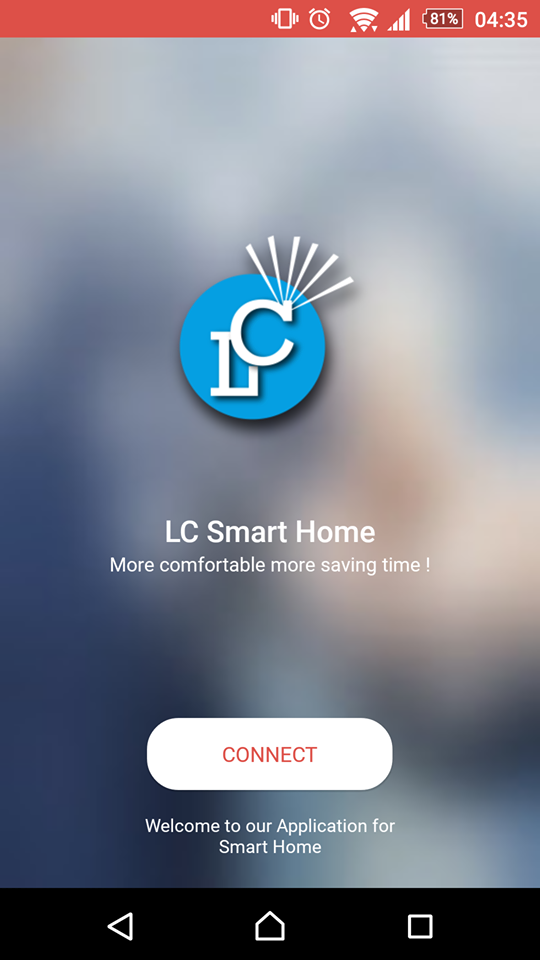
The XML files are charged only to do the graphics interfaces <<GUI>>

But it’s also possible to do the graphics with the Java programming language too,

If we use the XML file we can references the components needed in the XML file by

References.

‹‹@id/name\_Of\_Compenant››, The GUI resources can be found in the principal

Directory ‹‹res/layout›› and ‹‹res/values››. And this is an example of GUI defined by Xml file: Using this code that’s what do we get on the screen 

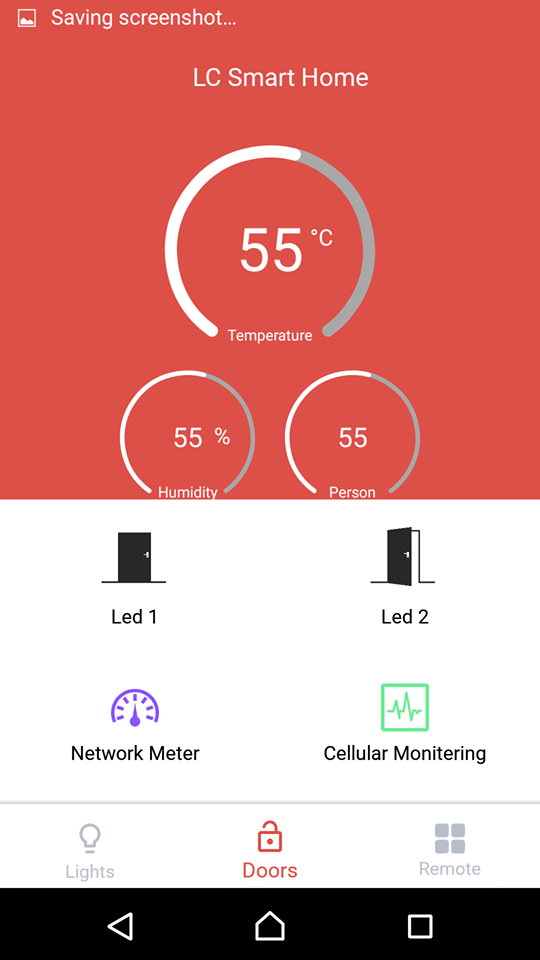
**3.3.3.2 Main user interface**

The main user interface contain 3 fragments which are divided to

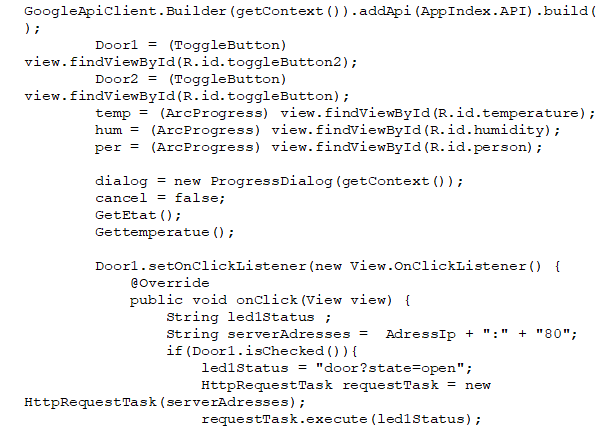
Door fragment, Led’s Fragment, Remote control fragment , after opening the app the focused fragment is the led’s fragment , any android screen need two things to be developed the Xml File and the Java source coding file , of course Xml for the GUI and java for the programming and this is what we’ve done with the three fragment :



XML Doors Fragment



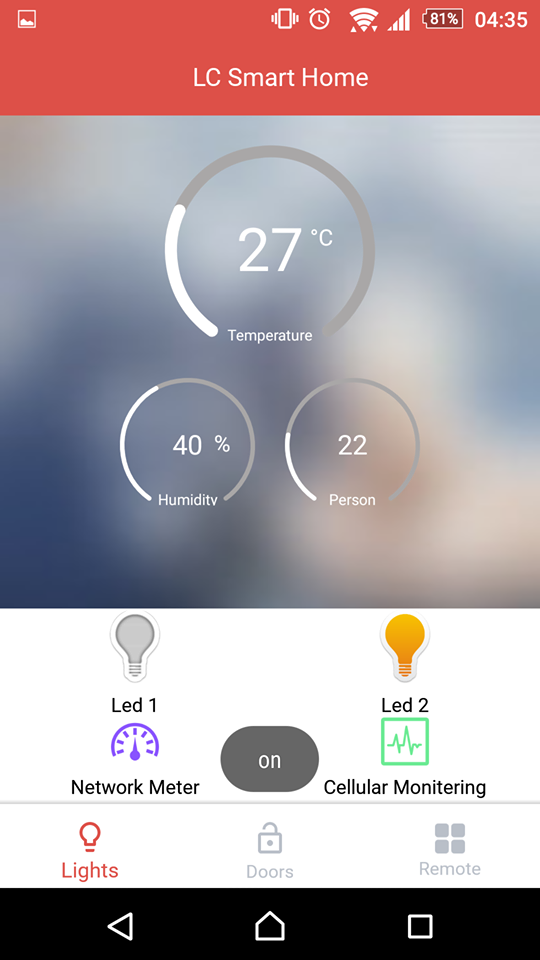
Doors\_Fragment



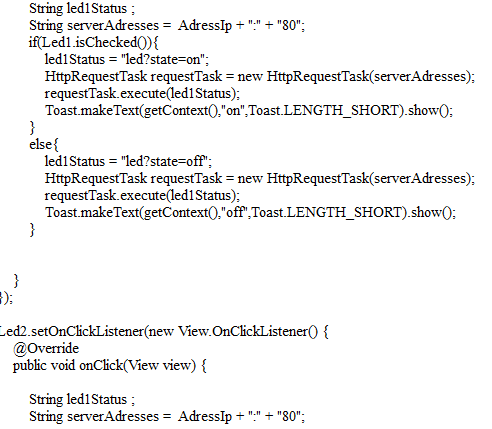
Java code example for Doors



Leds Xml source code



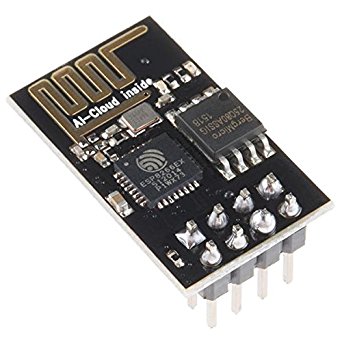
Leds Fragment



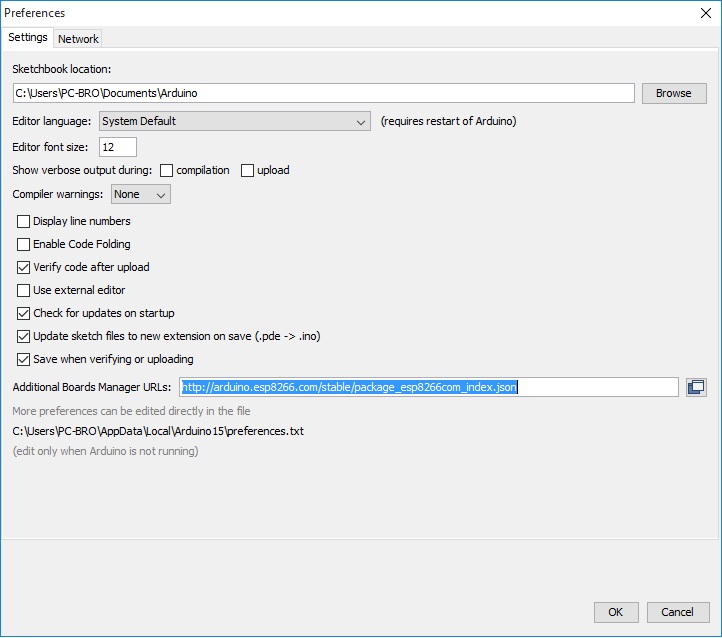
Leds Java Code

**3.3.4 ESP 8266 coding**

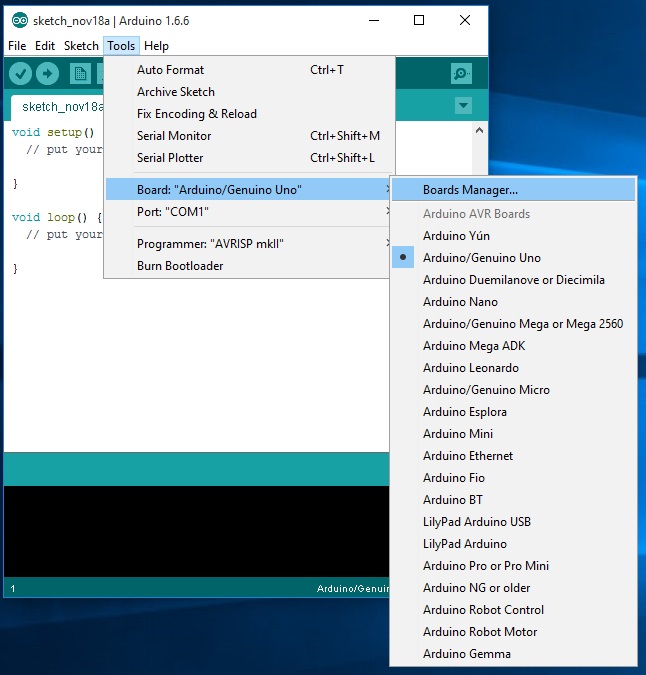
**3.3.4.1 Preparing IDE and ESP8266**

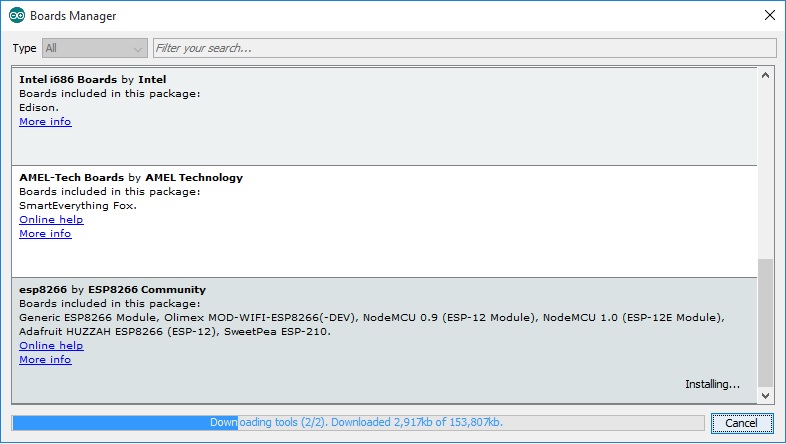
****

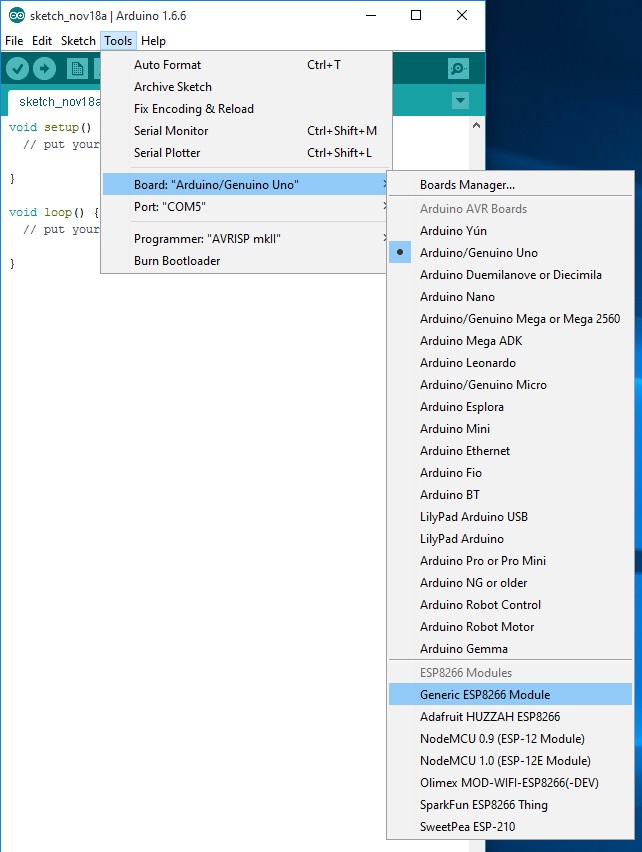
To program the Esp 8266 we need to work with Arduino IDE and it’s for arduino so to do that we need to add some library to the IDE so we can run any code simply on the ESP via the arduino IDE

****

Then we go to board managers and select the ESP 8266 and start downloading







Finally we select the ESP 8266 module and start programming

**3.3.4.2 Writing a program to the ESP8266**

Flashing a program to the ESP8266 is a bit more annoying than flashing an Arduino. When flashing the arduino, all you have to do is press the reset button and release while you upload a program (or even not doing anything if you have FTDI such as in arduino UNO,MEGA) and the arduino will start uploading. With the ESP8266 you have to reset the micro-controller and start it in flashing mode using the GPIO0-to-Ground already mentioned in some older posts. If you don't have any ESP8266 development board and only the module itself I advice building the next circuit Which will make your life easier.

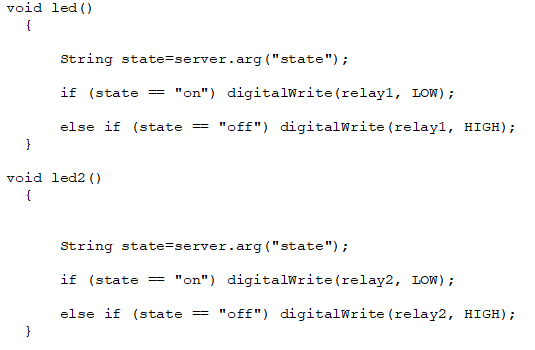
The sketch shows the formal way to connect ESP8266 to FTDI with a voltage regulator - However, two more buttons have been added. The right button, when pressed, connects the RESET pin to the ground and when it is released, connects the RESET pin to the VCC through a pull-up resistor. The left button, when pressed, connect GPIO0 to the ground. Using this two buttons you can do all the tasks you need with the ESP8266:

* Working on normal mode - Both buttons are released.
* Resetting the ESP8266 - Press the reset button and release.
* Start in flash mode - Press both buttons, release the reset button and then release the GPIO0 button.

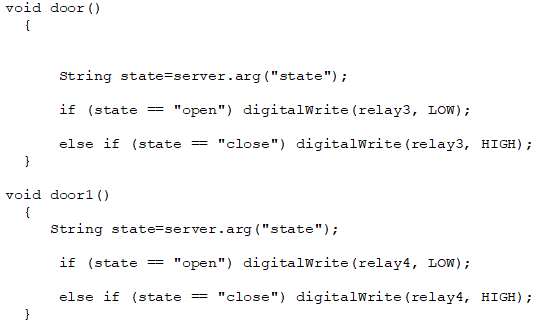
And that the way how you write your program to the ESP 8266 with the Arduino IDE .

**3.3.4.3 Creating our web server**

To reach the best way to control the led’s and the door’s with the other components we create a data base to store the state of led’s and doors, temp and humidity so there will be less pressure on the ESP8266 microchip because if it keep running there will be many bugs as the microchip don’t handle that much of operations, and there is some of the source code



That’s the function who control led’s

That’s the function who control doors

**3.3.5 Project**

To reach our goals we did merge the two application so we can control that ESP8266 with an android application the work was done by 100% but for reason of time and resources we couldn’t expend our project to much level higher we wanted to add more functionality to the project we wanted to test it with real door but we couldn’t get the resources we try it with a lamp and he did work properly





**3.4 Library’s**

To do that work we needed a lot of library’s on the arduino IDE and android studio here are the library’s:

**ESP library’s**

{ IRremoteESP8266.h

IRsend.h

IRrecv.h

IRutils.h

ir\_Daikin.h

ir\_Fujitsu.h

ir\_Gree.h

ir\_Haier.h

ir\_Kelvinator.h

ir\_Midea.h

ir\_Toshiba.h

ESP8266WiFi.h

WiFiClient.h

PubSubClient.h

ESP8266WebServer.h

ESP8266mDNS.h

DHT.h}

**Android Library’s**

{ com.github.lzyzsd.circleprogress.ArcProgress;

com.github.lzyzsd.circleprogress.CircleProgress;

com.google.android.gms.appindexing.AppIndex;

com.google.android.gms.common.api.GoogleApiClient;

design.ws.com.cleanupapplication.R;

}

Those are the library’s that we did include in our project we did mention them because we did search for them they are not in base application.

**Conclusion**

Smart homes will one day be the way all homes are lived in. This however will take time. Similar to the introduction of electricity in the turn of the century, the smart home business needs time to grow and mature before it becomes main-stream. There are, however, many benefits to owning a smart home due to the security and convenience it can provide. As more and more items, such as alarm systems, are introduced to the home, and items begin to work together, the rise of smart homes is definite.