**CHAPTER 2**

**LITERATURE REVIEW**

**2.0 INTRODUCTION**

The term Auction extracted from the Latin root auctus which means “I augment”. It is a  
process of purchasing and selling items through the bid and selling to the bidder who  
makes the highest offer. Usually the bidders that participate in the same auction item will  
increase the price against each other. The increasing prices continues until it reaches the  
highest price by the bidder. Many types of auction exist; each of them have their rules  
and conditions. The auction is a fact which items goes to the highest bidder. By publishing the items to the audience an auction may have a wide range of bids which make the  
item surprisingly goes to the very high price.

The Auction sales are held in the auction houses or in specific places. The people who  
wish to sell their items by auctioning take part in auction houses and publish their items  
for sale. Also, the buyers have to exist in the auction places which takes the time to arrive.  
In the auction sales, the seller has to pay some amount to the auctioneer who is managing  
the auction by accepting offering price and declaring items sold.

**2.1 HISTORY OF AUCTION**

The history of auction returns back to 500 B.C as stated by Herodotus. In Babylon where  
annually women were sold based on auction and being married to the person who has  
bought her. Women with less pretty were sold with returning money back to the bidder.  
The bids of these auctions were not mentioned whether they were ascending or descending.  
The Roman Empire used auctions to dispose of debt. One example is selling the own  
home‘s furniture by Marcus Aurelius to pay off debts.

The important historical auction happened in the year 193 A.D when the whole of Roman  
Empire was placed on auction block after being dismissed. In China also auction was  
used by Buddhist friars to put up money for temple creation. After that it became usual  
to auction of the possessions of deceased friars for this purpose.

The auction sales also existed in England, the London newspaper often notified the auctioning at coffeehouses and pubs everywhere in London in the late 17th century. The large  
auction houses were built in the early 18th century. Sotheby's was created in 1744 and  
Christie's was created in 1766.

In America auction sales appeared in the south when slaves were sold in auction. Often  
times, the merchandise owner did not allow to reveal himself because the present social  
customs might not view favorably to the auction which affect the auction negatively.  
In Netherland and Germany auction has also been used, the auctioning used in Netherland  
for selling fruits and vegetables return back to year 1887. In Germany fish was sold by  
auctioning**.**

Auction of art has been there since the 17th century. It has even become an important source of  
income to some companies. According to The New York times article (March 4 2016),  
companies that engage in art based lending hope to attract more collectors. According to  
Sotheby’s report, others have made up to even $11.2 million losses. Due to these problems,  
many websites and softwares and systems have been developed for doing online art auction.  
Although these sites have tried to solve all the problems, they cannot exhaust everything. Below  
is just an overview of what they have done so far.

**2.2 TYPES OF ONLINE AUCTIONS**

There are many different types of online auctions, the most important ones are:

**2.2.1 ENGLISH AUCTION**

The bids of English auction are declared either by auctioneer or by buyer. It is the  
most common type of online auction due to the easiness of use among other types.  
The format of bid auction is ascending. The bids are open for all to see. The bidder  
who makes the highest bid price is the winner. The online English auction is preferable by many people. Because it uses a simple mechanism that people find it familiar. Online English auction has become more popular than traditional English auction due to the fact that traditional English auction needs the physical existence of the bidders.

**2.2.2 DUTCH AUCTION**

Dutch auction begins in high price and the price decreases orderly to the price  
that become acceptable by the bidder. It is the reverse of English auction which is ascending the price. The Dutch auction commonly uses a uniform-price rule in a single unit auction. Mostly the items that put in online Dutch auction ends with the higher price than their first price which makes the bidder feel remorse about participating in Dutch auction.

**2.2.3 BIDDING FEE AUCTION**

In this kind of auction all participants must pay for bid but only one participant will win who makes the highest bid price. The participants buy the penny for makings bids instead of paying real money. The auction website's owner makes the money via buying the penny bids by bidders and actual value of final price.

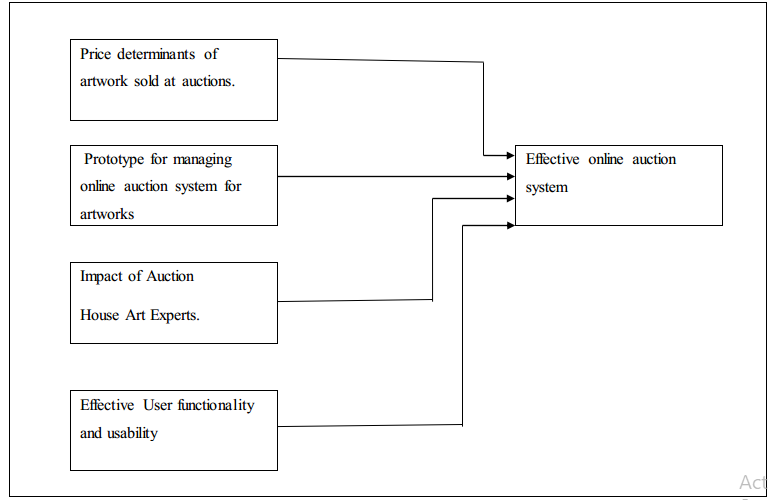
**2.2.4 FIRST-PRICE SEALED-BID**

The variation between this kind of auction and English auction is in this type of auction the bidders cannot see the offers made by another bidder. It will remain confidentially until the end of auction. While in English auction the bids are public and viewable by all bidders.

**2.2.5 SECOND-PRICE SEALED-BID**

It is known as Vickery auction too. It is very similar to first bid auction and it is working in the same way. The only difference between this type of auction and first bid auction is that the bidder with the highest bid who wins will pay the second highest bid instead of the bid made by him.

**2.3 THEORETICAL REVIEW/ CONCEPTUAL FRAMEWORK**



**figure 2.1 Conceptual framework**

**2.3.1 PRICE DETERMINANTS OF ARTWORK SOLD AT AUCTIONS**

The prices for art at auctions are determined by supply and demand. The interaction of supply of  
artwork with the demand from art investors produces the market clearing price at art auctions.  
This price can then be related to a set of characteristics, or price determinants. The price  
determinants of artworks at auctions are based on the hedonic pricing theory of heterogeneous  
goods, which in its own turn is grounded on consumer theory. In the case of the art market, an  
artwork can be interpreted as a combination of certain aesthetic features. It is the presence of  
these features that give the non-pecuniary utility, and when purchasing a piece of art, one is  
paying for these aesthetic and other attributes. Theoretically, there is no given set of art price  
determinants. Empirical studies rather employ the data available, which is usually a certain number of attributes of I ) the art work itself, ii) the artist, as well as iii) time and place of sale.  
Some of the price determinants used can be justified by an underlying economic reasoning of  
demand-supply interaction. Others, such as time and place related characteristics, are included  
for control, to account for demand. It is only by the use of online auction system for art that the  
quality and aesthetic value of art can be proven.

**2.3.2 Impact of Auction House Art Experts.**

The art experts at auction houses who evaluate artwork and give price estimate ranges can be  
compared to stock price analysts. Their price estimates can be seen as price forecasts. More  
often, pre-sale estimates provided by the auction house art experts serve as an anchor for actual  
realized prices during the auctions (Beggs and Graddy, 2010). Since the early 1970’s almost  
every art auction house provides an expert appraisal of each item that is put up for auction. Art  
auction houses typically have a team of art experts specialized in various types, movements and  
styles of art. These art experts make their appraisals based on their expertise and experience.  
They are aware of the past and current trends and monitor the art market closely. Each artwork  
receives a low estimate, as well as, a high estimate and the market value is estimated to lie  
between these two numbers. According to the general auction theory, the auctioneers are better  
off providing honest and truthful estimates (Milgrom and Weber, 2012).. The vast majority of  
the extant empirical studies focus on the forecasting accuracy of the price estimates that the  
auction house art experts give. Ashenfelter (1989) and Louargand and McDaniel (1991) are some  
of the first studies to look at the correlation between the auction house art experts’ estimates and  
realized prices. Both of the studies, using different, but small, datasets, conclude that the price  
estimates are very close to being unbiased, as they are highly correlated to realized prices. More  
recent studies that have confirmed the above findings using larger datasets and better methods  
include McAndrew, Smith and Thompson (2012). Ekelund, Ressler and Watson (2012), on the  
other hand, an upward bias in the auction house art expert estimates using a dataset of LatinAmerican art. Likewise, Mei and Moses (2013) show that there is a consistent upward bias in price estimates for expensive pieces of American art. Nevertheless, it is important to control for  
the impact of the auction house art expert’s appraisal of the artworks value. Most of the studies,  
such as the ones above, focus mainly on either the average of the high and low estimates or just  
one of them at a time. An exception is Mei and Moses (2013), where the estimate spread is  
included as well. Their finding is that art returns increase with the relative spread. Unfortunately,  
this finding is left without an explanation or justification. It is through the online auction system  
for art that some of the above biasness will be able to be done away with.

**2.3.3 PROTOTYPE FOR MANAGING ONLINE AUCTION SYSTEM**

The first ever made prototype for online auction was developed in 1995 by a research scientist E.  
skorve at NR. The auction system was implemented using common 3-tier architecture. All the  
information about the bids placed can be inserted in to the database at any time before the  
auction starts. This includes, the type of art that is either is a painting sculpture etc. when an  
auction is running information about the bidding process is also stored in the database. This  
eliminates two problems. Firstly, problems arising from simultaneous bids are eliminated since  
serializing access to the database tables is handled by the database engine itself. Secondly, since  
all user interaction is longed and time-stamped, a detailed post- analysis of bidder behavior can  
easily be performed, allowing tuning of the auction parameters and thus resulting in an improved  
auction model. The middle tier is the co-coordinator of the auction. It communicates with the database, inserting new bids for auction. It handles connection requests from bidders, registering them and their subsequent bids in to the database. It is responsible for broadcasting events to all participants such as information about the newest arts (bids)/this application is implemented as a standalone web based application. It communicates with database through xampp (php).

**2.3.4 TEST EFFECTIVE USER FUNCTIONALITY AND USABILITY**

According to (genesis framework, 2016) in usability testing basically the testers tests the ease  
with which the user interfaces can be used. It tests that whether the application or the product  
built is user friendly or not. It also reveals whether user feel comfortable with the application ar  
website according to different parameters. Usability testing tests how easy it is to use the system,  
how easy it is to learn the software and how convenient is the system to the end user.

**2.4 EXISTING ONLINE AUCTION SYSTEMS**

After knowing types of online auctions, it is a good thing to investigate about current  
online auction systems around the net. Providing their functionalities. Two famous auction systems are considered for this research eBay.com and huuto.net.

**2.4.1 EBAY.COM**

eBay is the world's largest online marketplace which let people to trade locally and worldwide. The people are able to buy and sell on eBay through online auction formats, which is also called auction-style listings. Or direct buying through 'buy it now’. eBay was founded by Pierre Omidyar in 1995. eBay acts solely as a listing-agent. Auctions typically last 3-7 days from the placement of an art based on the duration the seller selects at the time the bid is placed .the sellers and the bidder  
must fill in the registration form with email address and has to be used in a reply and  
conformation sequences. The user uses a ‘bid-button’ to send to eBay information about the  
maximum amount he/she is willing to pay for the item. The amount is kept secret from other  
participants and eBay is bidding on behalf of the person as other bidders increase their bids.  
eBay automatically terminates the bidding at the end of the auction, and notifies the seller and  
the highest bidder via email. Then the buyer and the seller should contact each other within three  
business days of the end of the auction.

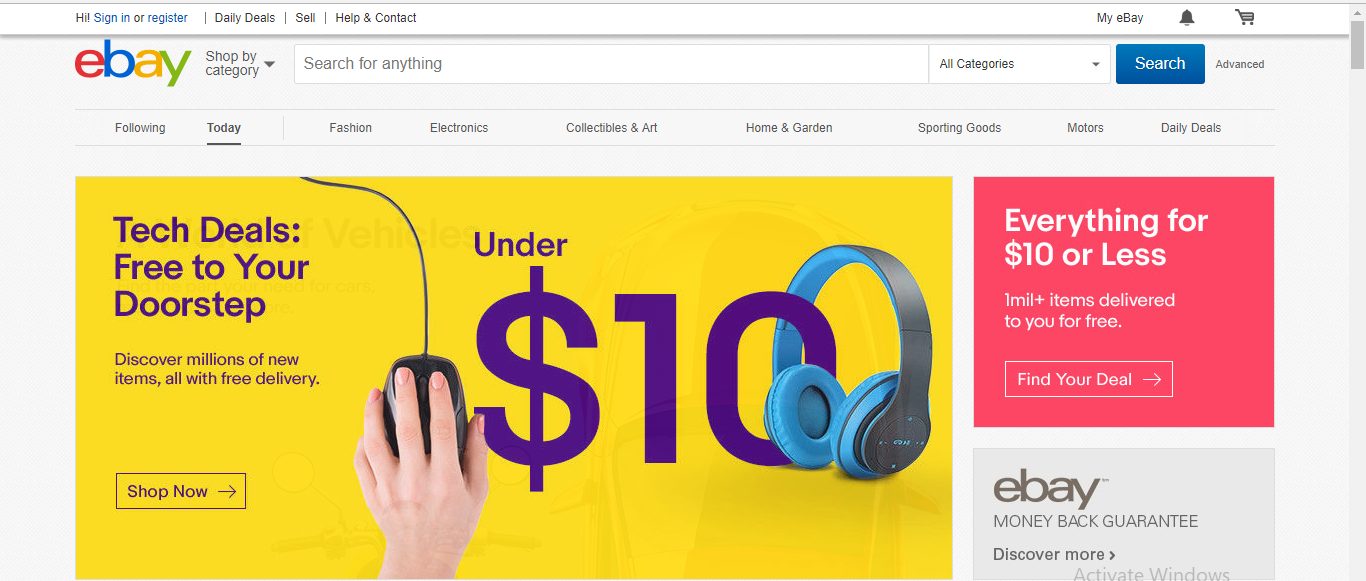


Figure 2.2 : Ebay.com

**2.4.2 HUUTO.NET**

Huuto.net is a Finnish online auction much the same as the global eBay. It was founded  
in 1999 by Lari Lohikoski in Helsinki. The name “huuto” is extracted from "huutokauppa" which means 'auction' in Finnish language.

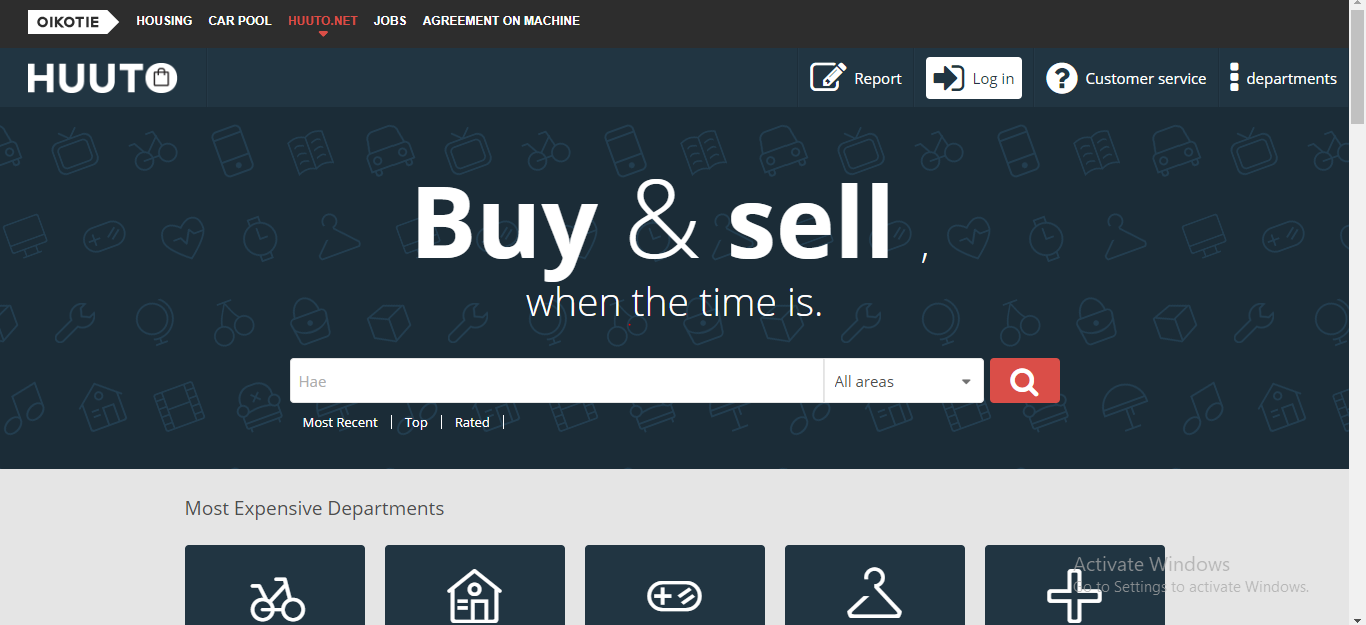


Figure 2.3: Huuto.net

**2.4.3 THE FUNCTIONALITIES OF THE SYSTEM**Table below shows the main functionalities between both systems.  
TABLE 1. Functionalities of the systems.

|  |  |  |
| --- | --- | --- |
| User stories | Ebay.com | Huuto.net |
| Login | Χ | Χ |
| Register | Χ | Χ |
| Home page | Χ | Χ |
| Chat | Χ |  |
| Own page | Χ | Χ |
| Place a bid | Χ | Χ |
| Upload item | Χ | Χ |
| feedback | Χ | Χ |
| Change language |  |  |
| Search by category | Χ | Χ |
| Excluding a word | Χ | Χ |
| By city | Χ |  |
| Message | Χ |  |
| Help | Χ | Χ |
| Shopping cart | Χ |  |
| Sitemap | Χ |  |
| Administrator | ? | ? |

As shown in the table 1 above, the two systems allow the user possibility to register, to  
login to the website and have a home page. Both websites also have a personal page,  
where each user can monitor the status of their auctions or of their bids and changing their  
personal information. Another function of both websites is having an item page, a page  
that represents each item on auction (with a name, a description, a photo etc.). The search  
functionality is also very important: in addition to a normal keyword search, both websites  
offer also the possibility to search excluding a given word, search in a given category.  
Huuto.net offers also to search for auctions regarding a given city. The two systems give  
also the possibility to place a bid, to upload an auction and have also some help pages.  
The feedback functionality is the important function which exists in both sites, it gives  
the users opportunity to leave their feedback about their experience when they are buying  
or selling something. On eBay, the user also has the ability to chat and sends a message  
to another user. Another important feature on eBay is shopping cart which is allowing the  
buyers to compare and see the final cost of the item before checkout.  
The sitemap is another important functionality on eBay to make easier navigation through  
the website. None of the two websites has the ability to change language (huuto.net is in  
Finnish, while eBay.com is in English but has different versions in multiple languages for  
example ebay.es is in Spanish language and ebay.de is in Deutsch language and so on).

**2.5 THEORETICAL TECHNOLOGY FOR DEVELOPMENT**

**2.5.1 C#, Programming Language**

C# [(/si: ʃɑːrp/)](https://en.wikipedia.org/wiki/Help:IPA/English) is a [multi-paradigm programming language](https://en.wikipedia.org/wiki/Multi-paradigm_programming_language) encompassing [strong typing](https://en.wikipedia.org/wiki/Strong_typing), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [declarative](https://en.wikipedia.org/wiki/Declarative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming), [generic](https://en.wikipedia.org/wiki/Generic_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) ([class](https://en.wikipedia.org/wiki/Class_(computer_science))-based), and [component-oriented](https://en.wikipedia.org/wiki/Component-based_software_engineering) programming disciplines. It was developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) within its [.NET](https://en.wikipedia.org/wiki/.NET_Framework) initiative and later approved as a standard by [Ecma](https://en.wikipedia.org/wiki/Ecma_International" \o "Ecma International) (ECMA-334) and [ISO](https://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO/IEC 23270:2006). C# is one of the programming languages designed for the [Common Language Infrastructure](https://en.wikipedia.org/wiki/Common_Language_Infrastructure).

C# is a general-purpose, object-oriented programming language. Its development team is led by [Anders Hejlsberg](https://en.wikipedia.org/wiki/Anders_Hejlsberg). The most recent version is C# 7.2, which was released in 2017 along with [Visual Studio](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio) 2017 version 15.5.

**2.5.1.1 DESIGN GOAL**

The ECMA standard lists these design goals for C#:

* The language is intended to be a simple, modern, general-purpose, [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) language.
* The language, and implementations thereof, should provide support for software engineering principles such as [strong type](https://en.wikipedia.org/wiki/Strong_type)checking, array [bounds checking](https://en.wikipedia.org/wiki/Bounds_checking), detection of attempts to use [uninitialized variables](https://en.wikipedia.org/wiki/Uninitialized_variable), and automatic [garbage collection](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). Software robustness, durability, and programmer productivity are important.
* The language is intended for use in developing [software components](https://en.wikipedia.org/wiki/Software_components) suitable for deployment in distributed environments.
* Portability is very important for source code and programmers, especially those already familiar with [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B).
* Support for [internationalization](https://en.wikipedia.org/wiki/Internationalization_and_localization) is very important.
* C# is intended to be suitable for writing applications for both hosted and [embedded systems](https://en.wikipedia.org/wiki/Embedded_system), ranging from the very large that use sophisticated [operating systems](https://en.wikipedia.org/wiki/Operating_system), down to the very small having dedicated functions.

**2.5.1.2 HISTORY**

During the development of the .NET Framework, the [class libraries](https://en.wikipedia.org/wiki/Base_Class_Library) were originally written using a [managed code](https://en.wikipedia.org/wiki/Managed_code) compiler system called *Simple Managed C* (SMC). In January 1999, [Anders Hejlsberg](https://en.wikipedia.org/wiki/Anders_Hejlsberg) formed a team to build a new language at the time called Cool, which stood for "[C-like](https://en.wikipedia.org/wiki/C-like) Object Oriented Language".[[19]](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)#cite_note-computerworld2008-21) Microsoft had considered keeping the name "Cool" as the final name of the language, but chose not to do so for trademark reasons. By the time the .NET project was publicly announced at the July 2000 [Professional Developers Conference](https://en.wikipedia.org/wiki/Professional_Developers_Conference), the language had been renamed C#, and the class libraries and [ASP.NET](https://en.wikipedia.org/wiki/ASP.NET) runtime had been ported to C#.

Hejlsberg is C#'s principal designer and lead architect at Microsoft, and was previously involved with the design of [Turbo Pascal](https://en.wikipedia.org/wiki/Turbo_Pascal), [Embarcadero Delphi](https://en.wikipedia.org/wiki/Embarcadero_Delphi) (formerly CodeGear Delphi, Inprise Delphi and Borland Delphi), and [Visual J++](https://en.wikipedia.org/wiki/Visual_J%2B%2B). In interviews and technical papers he has stated that flaws[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)] in most major programming languages (e.g. [C++](https://en.wikipedia.org/wiki/C%2B%2B), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [Delphi](https://en.wikipedia.org/wiki/Embarcadero_Delphi), and [Smalltalk](https://en.wikipedia.org/wiki/Smalltalk)) drove the fundamentals of the [Common Language Runtime](https://en.wikipedia.org/wiki/Common_Language_Runtime) (CLR), which, in turn, drove the design of the C# language itself.

Since the release of C# 2.0 in November 2005, the C# and Java languages have evolved on increasingly divergent trajectories, becoming two very different languages. One of the first major departures came with the addition of [generics](https://en.wikipedia.org/wiki/Generic_programming) to both languages, with vastly different implementations. C# makes use of [reification](https://en.wikipedia.org/wiki/Reification_(computer_science)) to provide "first-class" generic objects that can be used like any other class, with code generation performed at class-load time.[[25]](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)#cite_note-27) Furthermore, C# has added several major features to accommodate functional-style programming, culminating in the [LINQ](https://en.wikipedia.org/wiki/Language_Integrated_Query) extensions released with C# 3.0 and its supporting framework of [lambda expressions](https://en.wikipedia.org/wiki/Lambda_expressions), [extension methods](https://en.wikipedia.org/wiki/Extension_method), and [anonymous types](https://en.wikipedia.org/wiki/Anonymous_type).

These features enable C# programmers to use functional programming techniques, such as [closures](https://en.wikipedia.org/wiki/Closure_(computer_science)), when it is advantageous to their application. The LINQ extensions and the functional imports help developers reduce the amount of [boilerplate code](https://en.wikipedia.org/wiki/Boilerplate_code) that is included in common tasks like querying a database, parsing an xml file, or searching through a data structure, shifting the emphasis onto the actual program logic to help improve readability and maintainability.

**2.5.1.3 NAME**

The name "C sharp" was inspired by musical notation where a [sharp](https://en.wikipedia.org/wiki/Sharp_(music)) indicates that the written note should be made a [semitone](https://en.wikipedia.org/wiki/Semitone) higher in [pitch](https://en.wikipedia.org/wiki/Pitch_(music)). This is similar to the language name of [C++](https://en.wikipedia.org/wiki/C%2B%2B), where "++" indicates that a variable should be incremented by 1. The sharp symbol also resembles a [ligature](https://en.wikipedia.org/wiki/Typographic_ligature) of four "+" symbols (in a two-by-two grid), further implying that the language is an increment of C++.

**2.5.1.4 SYNTAX**

The core syntax of C# language is similar to that of other C-style languages such as C, C++ and Java. In particular:

* Semicolons are used to denote the end of a statement.
* [Curly brackets](https://en.wikipedia.org/wiki/Curly_brackets) are used to group statements. Statements are commonly grouped into methods (functions), methods into classes, and classes into [namespaces](https://en.wikipedia.org/wiki/Namespaces).
* Variables are assigned using an [equals sign](https://en.wikipedia.org/wiki/Equals_sign), but compared using [two consecutive equals signs](https://en.wikipedia.org/wiki/%3D%3D).
* [Square brackets](https://en.wikipedia.org/wiki/Square_brackets) are used with [arrays](https://en.wikipedia.org/wiki/Array_data_structure), both to declare them and to get a value at a given index in one of them.

**2.5.1.5 EXAMPLES**

The following is a very simple C# program, a version of the classic "[Hello world](https://en.wikipedia.org/wiki/Hello_world)" example:

using System;

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello, world!");

}

}

What will display on the program is:

Hello, world!

Each line has a purpose:

**Using System;**

The above line of code tells the compiler to use System as a candidate prefix for types used in the source code. In this case, when the compiler sees use of the Console type later in the source code, it tries to find a type named Console, first in the current assembly, followed by all referenced assemblies. In this case the compiler fails to find such a type, since the name of the type is actually System.Console. The compiler then attempts to find a type named System.Console by using the System prefix from the using statement, and this time it succeeds. The using statement allows the programmer to state all candidate prefixes to use during compilation instead of always using full type names.

**Class Program**

Above is a [class](https://en.wikipedia.org/wiki/Class_(computer_science)) definition. Everything between the following pair of braces describes Program.

**static void main(string[] args)**

This declares the class member method where the program begins execution. The .NET runtime calls the Main method. (Note: Main may also be called from elsewhere, like any other method, e.g. from another method of Program.) The [static keyword](https://en.wikipedia.org/wiki/Method_(computer_programming)#Static_methods) makes the method accessible without an instance of Program. Each console application's Main entry point must be declared static. Otherwise, the program would require an instance, but any instance would require a program. To avoid that irresolvable [circular dependency](https://en.wikipedia.org/wiki/Circular_dependency), C# compilers processing [console applications](https://en.wikipedia.org/wiki/Console_application) (like that above) report an error, if there is no static Main method. The void keyword declares that Main has no [return value](https://en.wikipedia.org/wiki/Return_value).

**Console.WriteLine("Hello, world!");**

This line writes the output. Console is a static class in the System namespace. It provides an interface to the standard input, output, and error streams for console applications. The program calls the Console method WriteLine, which displays on the console a line with the argument, the string "Hello, world!".

**2.5.1.6 C# Version**

**C# Version 1**

C# version 1 really did look an awful lot like Java.  As [part of its stated design goals for ECMA](http://feeldotneteasy.blogspot.com/2011/01/c-design-goals.html), it sought to be a “simple, modern, general purpose object-oriented language.”  At the time, it could have done worse thank looking like Java in order to achieve those goals.

But if you looked back on C# 1.0 now, you’d find yourself a little dizzy.  It lacked the built in async capabilities and some of the slick functionality around generics that we take for granted.  As a matter of fact, it lacked generics altogether.  And [Linq](https://msdn.microsoft.com/en-us/library/bb308959.aspx)?  Nope.  That would take some years to come out.

**C# Version 2**

Now things start to get interesting.  Let’s take a look at some major features of C# 2.0, released in 2005, along with Visual Studio 2005.  (Check out [the book by NDepend creator Patrick Smacchia](http://www.ndepend.com/practicalbook) about .NET 2.0.)

* [Generics](https://www.tutorialspoint.com/csharp/csharp_generics.htm)
* [Partial types](https://www.dotnetperls.com/partial)
* [Anonymous methods](https://www.tutorialspoint.com/csharp/csharp_anonymous_methods.htm)
* [Nullable types](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/nullable-types/)
* [Iterators](https://msdn.microsoft.com/en-us/library/65zzykke(v=vs.100).aspx)
* [Covariance and contravariance](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/covariance-contravariance/)

While Microsoft may have started with a pretty generic object-oriented language, C# Version 2 changed that in a hurry.  Once they had their feet under them, they went after some serious developer pain points.  And they went after them in a big way.

With generics, you have types and methods that can operate on an arbitrary type while still retaining type safety.  So, for instance, having a List<T> lets you have List<string> or List<int>  and perform type safe operations on those strings or ints while you iterate through them.  This certainly beats creating ListInt inheritors or casting from Object for every operation.

**C# Version 3**

C# Version 3 came in late 2007, along with Visual Studio 2008, though the full boat of language features would actually come with C# Version 3.5.  And what a version this proved to be.  I would go so far as to say that this established C# as a truly formidable programming language.  Let’s take a look at some major features in this version.

* [Auto implemented properties](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/auto-implemented-properties)
* [Anonymous types](http://www.c-sharpcorner.com/UploadFile/ff2f08/anonymous-types-in-C-Sharp/)
* [Query expressions](https://docs.microsoft.com/en-us/dotnet/csharp/linq/query-expression-basics)
* [Lambda expression](http://www.daedtech.com/introduction-to-c-lambda-expressions/)
* [Expression trees](https://blogs.msdn.microsoft.com/charlie/2008/01/31/expression-tree-basics/)
* [Extension methods](https://www.codeproject.com/Tips/709310/Extension-Method-In-Csharp)

**C# Version 4**

C# Version 4 would have had a difficult time living up to the groundbreaking status of version 3.  With version 3, Microsoft had moved the language firmly out from the shadow of Java and into prominence.  The language was quickly becoming elegant.

The next version did introduce some cool stuff, though.

* [Dynamic binding](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/dynamic)
* [Named/optional arguments](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/named-and-optional-arguments)
* [Generic covariant and contravariant](https://docs.microsoft.com/en-us/dotnet/standard/generics/covariance-and-contravariance)
* [Embedded interop types](https://stackoverflow.com/questions/20514240/whats-the-difference-setting-embed-interop-types-true-and-false-in-visual-studi)

**C# Version 5**

With C# Version 5, Microsoft released a very focused version of the language.  They put nearly all of their effort for that version into another pretty groundbreaking language concept.  Here is the major features list.

* [Asynchronous members](https://msdn.microsoft.com/library/hh191443(vs.110).aspx)

**C# Version 6**

With versions 3 and 5, Microsoft had done some pretty impressive stuff in an OO language.  (Version 2 did as well, but they were fast following Java with those language features.)  With version 6, they would go away from doing a dominant killer feature and instead release a lot of features that delighted users of the language.  Here are some of them.

* [Static imports (a la Java)](http://geekswithblogs.net/BlackRabbitCoder/archive/2015/04/16/c.net-little-wonders-static-using-statements-in-c-6.aspx)
* [Exception filters](https://www.thomaslevesque.com/2015/06/21/exception-filters-in-c-6/)
* [Property initializers](http://geekswithblogs.net/WinAZ/archive/2015/06/30/whatrsquos-new-in-c-6.0-auto-property-initializers.aspx)
* [Expression bodied members](https://lostechies.com/jimmybogard/2015/12/17/c-6-feature-review-expression-bodied-function-members/)
* [Null propagator](https://davefancher.com/2014/08/14/c-6-0-null-propagation-operator/)
* [String interpolation](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/interpolated-strings)
* [nameof operator](https://stackoverflow.com/questions/31695900/what-is-the-purpose-of-nameof)
* [Dictionary initializer](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/how-to-initialize-a-dictionary-with-a-collection-initializer)

**C# Version 7**

Finally, we arrive at C# version 7.  That’s the current version as of the writing of this post.  This has some evolutionary and cool stuff in the vein of C# 6, but without the compiler as a service.  Here are some of the new features.

* [Out variables](http://www.c-sharpcorner.com/article/out-variables-in-c-sharp-7-0/)
* [Tuples and deconstruction](https://www.thomaslevesque.com/2016/08/23/tuple-deconstruction-in-c-7/)
* [Pattern matching](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7#pattern-matching)
* [Local functions](http://www.infoworld.com/article/3182416/application-development/c-7-in-depth-exploring-local-functions.html)
* [Expanded expression bodied members](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7#more-expression-bodied-members)
* [Ref locals and returns](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7#ref-locals-and-returns)

**2.5.2 TRANSACT-SQL** (**T-SQL**)

It is [Microsoft](https://en.wikipedia.org/wiki/Microsoft)'s and [Sybase](https://en.wikipedia.org/wiki/Sybase)'s proprietary extension to the [SQL](https://en.wikipedia.org/wiki/SQL) (Structured Query Language) used to interact with [relational databases](https://en.wikipedia.org/wiki/Relational_database). T-SQL expands on the SQL standard to include [procedural](https://en.wikipedia.org/wiki/Procedural_programming) programming, [local variables](https://en.wikipedia.org/wiki/Local_variable), various support functions for string processing, date processing, mathematics, etc. and changes to the [DELETE](https://en.wikipedia.org/wiki/Delete_(SQL)) and [UPDATE](https://en.wikipedia.org/wiki/Update_(SQL)) statements.

Transact-SQL is central to using [Microsoft SQL Server](https://en.wikipedia.org/wiki/Microsoft_SQL_Server). All applications that communicate with an instance of SQL Server do so by sending Transact-SQL statements to the server, regardless of the user interface of the application. [Stored procedures](https://en.wikipedia.org/wiki/Stored_procedure) in SQL Server are executable server-side routines. The advantage of stored procedures is the ability to pass parameters.

**2.5.3 MICROSOFT SQL SERVER**

Microsoft SQL Server is a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft). As a [database server](https://en.wikipedia.org/wiki/Database_server), it is a [software product](https://en.wikipedia.org/wiki/Software_product) with the primary function of storing and retrieving data as requested by other [software applications](https://en.wikipedia.org/wiki/Software_application)—which may run either on the same computer or on another computer across a network (including the Internet).

Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many [concurrent users](https://en.wikipedia.org/wiki/Concurrent_user).

The history of Microsoft SQL Server begins with the first Microsoft SQL Server product – SQL Server 1.0, a 16-bit server for the [OS/2](https://en.wikipedia.org/wiki/OS/2) operating system in 1989 - and extends to the current day.

Milestones

* MS SQL Server for OS/2 began as a project to port [Sybase SQL Server](https://en.wikipedia.org/wiki/Sybase_SQL_Server) onto OS/2 in 1989, by [Sybase](https://en.wikipedia.org/wiki/Sybase), [Ashton-Tate](https://en.wikipedia.org/wiki/Ashton-Tate), and [Microsoft](https://en.wikipedia.org/wiki/Microsoft).
* SQL Server 4.2 for NT is released in 1993, marking the entry onto [Windows NT](https://en.wikipedia.org/wiki/Windows_NT).
* SQL Server 6.0 is released in 1995, marking the end of collaboration with [Sybase](https://en.wikipedia.org/wiki/Sybase); Sybase would continue developing their own variant of *SQL Server*, Sybase [Adaptive Server Enterprise](https://en.wikipedia.org/wiki/Adaptive_Server_Enterprise), independently of Microsoft.
* SQL Server 7.0 is released in 1998, marking the conversion of the source code from C to C++.
* SQL Server 2005, released in 2005, finishes the complete revision of the old Sybase code into Microsoft code.
* SQL Server 2017, released in 2017, adds Linux support for these Linux platforms: [Red Hat Enterprise Linux](https://en.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), [SUSE Linux Enterprise Server](https://en.wikipedia.org/wiki/SUSE_Linux_Enterprise_Server), [Ubuntu](https://en.wikipedia.org/wiki/Ubuntu_(operating_system)) & [Docker Engine](https://en.wikipedia.org/wiki/Docker_(software)" \o "Docker (software)).[[3]](https://en.wikipedia.org/wiki/Microsoft_SQL_Server#cite_note-3)

Currently

As of October 2017 the following versions are supported by Microsoft:

* SQL Server 2008
* SQL Server 2008 R2
* SQL Server 2012
* SQL Server 2014
* SQL Server 2016
* SQL Server 2017