This document is about how the constrol system of a automated greenhouse plant management system.

This project will try to use raspberry pi as hardware for running this project.

This project provide these abilities: the console provide command to load dll or cs file, provide the the command that can call command and can automatically call based on some kine of status? I don't know exactly how to do this.

To load dll or cs files, I will use Roslyn Compiler API to compile code in runtime. Because of that, I will need to manage all dll and cs files that be compiled in runtime as object. There is no requirement for them, this manager just need to know what being compiled at this time.

The big problem is how to provide the ability that function from compiled at runtime can be called in console. That mean, I need to have syntax for calling them, finding a way to provide data they need.

Let go to first problem, what syntax I want in console? I think these command can be divide into 2 regions: for managing the compiled in runtime files and for doing task.

For managing the compiled in runtime command, I think there are 2 commands: the one that load and the one that show the some kind of status.

With the one that load, along with loading something, it also need to write to console that file was loaded or not. So, I think, some command from console that compile some file will look like this:

Please compile file from [directory]

And in return, the console will answer like this:

Seccessfully compile case:

Successfully compile file from [directory]

Failure load case:

Fail to compile file from [directory].

Errors:

[Error] : [Error description]

That is the behavior and format of command that I want.

The next problem is format of comnad for calling some function. And it depends on how the code be designed. I don't know should I let the manager who use this project and call function from console can call any function or not.

I want to know all command can be executed, so I think some manager will be needed to control all of that. But the problem is, what will be the command type look like? It is obviously that if I want to accept multiple kind of input, then I will need a generic type. But is that truly answer? Let go back a little bit, and stay in that a little. So, what is a command? Is it some function of a type object? Or it is an object? What will be different of them? I don't think that there is a way for me to control a collection of command, try to run some function of it without knowing it function at compile time. If I can merge them into one interface, then I just need to know it that it is something that I know. So the first point that different of them is what I know about them at compile time. At compile time, I don't know exactly what function need to called with a command that is some function, otherwise, I know exactly what need to be called with a command that is a special object. The next side we need to see is what params command need to take to run. At compile time, I can't know exactly what command need. So if a command called from console with param (somehow), I need to convert that param to the param that command need, right? So it is obviously that I need to boxing that? The question here is, am I need to do that or more exactly, is that problem of command manager? What should take this dependency, the command or the manager? I think it is some kind of stupid if I let the command have to convert param to what it need. It is not sure that the command can know what it received, and that thing have the same meaning that the command cannot sure what should it do with that kind of object. The caller, maybe the manager, should convert it to match the command require. Because of that, I don't think if I want the command is some kind of function, it can do it. So, a command need to be a generic type. And that mean, we will have some kind of register pattern for manager these commands?

Let think of it a little bit. What we know is the command is some kind of object that inherited to some kind of design that define all command, right? So that mean that at compile time, we know all commands can do something, right? The only thing we don't know at compile time, is what is the thing that command need to run, right? So, from command propectives, how we can provide that information? Should we look to command that, if two commands need different type of param, they are different? Or even they have different require for type of param, the all the same? What is different between those choice? Different type of command mean as it called, and it go the same way as one type of command. The problem is here, a collection can only have a collection of specific type, not multiple kind of type. Another aspect need to be considred is how we demand that command do it mission. It mean that where should we provide information about the type that command need to run. With the case that commands different with each other if they have different of param type require, when we call it, or demand it do its function, we just need to say it? I mean, at the moment we call, the command have known exactly which type of param it need, so we don't need to provide it when we call it, right? But it don't go the same way with the situation when all command will be the same. When we need a command to do its function, it is obviously that we will need to provide the information about the type of command need when it run. So the question that, when we want to provide that information of param type for command? When we define it, or when we call it? Each command should know exactly what it need to run, I think that. Because of that, we will need to find a way to gather all different type of command to one collection.

So let think about it a litte bit. What is it need? So, I need to manage all command can be called from console, that mean I need some thing to contain all command, right? So, how to call a command? The caller need to providee name of the command, the parameter that it need to run, and that it. The name of command will be the name of class that defined that command, and the parameter it needed to run will be the parameter of the execute method. But, it is obviously that each command can have different requirement of parameter that it needed. Because of that, we can suppose that each command will be something that heave different type with each other, that mean we can suppose that each command have their own type. This mean if we need a container that can contain the object of multiple type. I'm not sure is this thing true if instead of object, if I contain the way to create the object. But also, I don't know exactly how to do this. Let think about this way. Even when I define the containter contain the way, I still need to care about what to return and what to passed for it. This thing is true with the container that contain object. In both way, there is no type that can be general command because each command require different param. Even when we ignore the param, with different return command type need to return, there will need several factory need to define for those command. So, the point is we have to also, ignore the type and the value of item in container. Because of that I think the container contain object will be better choice in this situation. I'm not sure about this, but at least I can hope with this kind of design, I can reuse code easier. So, I think the best way to do this is make the container become a register pattern. I'm not sure is it a good choice, but it is the only answer I have now. So, because we have to ignore the type, I think we willl have to boxing the value back to the type it needed. But what is the best way to do this? Also, I need to think if I create a container, can it be good as the DI Container provided by microsoft? I don't think so, so I think it is the best to use it.

Now the problem have been shifted to what is the best lifetime type for command object and how to know which command need to be executed. That make me think that implement only ICommand just not enough. A command need to know is it called or not. And a call mean something that passed from console to tell command run. Now it is the time to consider how problem should be looked. I mean, with somekind of input, how to find the correct command to execute? So, which one find the other? Is the command find themselves called or the called find the command need to called? Or in other word, would we passed the request to all command to know which command can answer the request or ignore what will answred the request, point some command that we feel comfort to answered the request? What make two approach different is what is the thing that consider can a command answer request or not. Is it command know it self should answer request or not, or the caller decide what command should be called to answer the request? Where should this information be defined? At this moment, I don't think that the caller know what command should be called is a good idea. Because it is the part that can't be expanded. This also mean that ICommand cannot have just an Execute method. Command need something else to tell the caller that it can or cannot do the request. So, I think a Answer method which return bool value will be our answer for this problem. Because all of these things, I will try to have some kind of design like "Chain of responsibility" pattern, but I don't want to let the command know each other, so it is something look like that pattern, but it isn't that pattern.

The only problem left how a command define that tell everyone that with a command input from console like "run", it can answer to that request or not. It is posible to have some design in that the command have same in require input but have different name. So, which type that make the command difference with each other will be another problem too. It is the request text that command will response too. So, a command will have these characteristics: the request text that come from console, the param type required of it execute method and maybe it name? I'm not sure about the last one, but the first and second one is something that isn't allowed to be ignored. With a generic command, it shouldn't have to care about how it need to answer a request from console, so I think I will need another interface that request command itself know can it answer a request or not.

All solved, the next problem will be how to write the core command of program: load command.

I think it will be easy task to do this. But the problem is this: I want the command like log command can auto called everytime a command called and I need to find somehow, when command execute and it need param to run, I have to provide it to the command. I will go with the second problem first.

Let look at the problem. What do we have here? We know that a command will have some kind of syntax. It need to specified an object to do it mission. All of this have to do through some kind of syntax from console. So the syntax that command know to identify something will be designed by it own. The real problem is how to specify a object, which we don't know is it created or not, in console. So, there are two problems here: how to create an object that needed and how to know how to provide it. There is always a clear solution for this type of problem: boxing. So what we care about the collection of parameters is their name and their value? So, just save their information of type and value, give them some unique name the save it some kind of dictionary. If I provide ability to create, I will also want abiltiy that can modify and delete. I can call it normally as variable in generic case, but in our specific case now, some name like status will be more suitable. Another problem I relise is that when you like, put a command to console, you need to transfer it to code and make it run. So I need some kind of tool that do this. Roslyn can do this too, so I think I don't need to worry about it.

Now is the problem of how to let command can be called when other command be called. Or in bigger picture, that mean command can be called automatically base on matched condition. Match request that defined in command is some kind of matched condition too. I need a little time to think about it. So, what is the caller? Is that command call other command, or when received a request, the manager will execute both command? What is the difference between them? So with the manager call both command to execute, the manager need to know exactly both of them, right? So, identify a command with a method, is it still good enough design? At least, in this situation, when receive a request manager need to find from all command is there can handle the request, it is hard to say it is posible to specify the command need to run with a request. In that situation, we donot know what command need to run with a request, right? It is posible if I change the design that the command when loaded by manager can be registered with request, but it maybe can, also I donnot know is this a good idea or not, multiple command have the same request. But is it a design that allow the command can know other command? When we now do not need the command need to know when is should be called in hard code but when it registered only, then we must ask ourself about how command is stored. Now I think it is little unclear when think about this problem. So, how a command can call other command? Perhap they may know each other in hard code? Perhap they use other command as initialized param to intialize? Perhap there is a mechanism of the manager that allow a command can register to let it run with other command? There is nothing to do if there is someway when command call other command with hard code. The manager simply can not manage that. But if a command require an other command to initialize, that mean the manager when create command object will have some mechanism to let it register too. So it is the time to decide which command should be called too when a command be called. How we see it fit will the answer of this problem. So, there are only two ways: at compile time or at runtime. At compile time, the manager have nothing to do. At runtime, the manager have to provide a way that let a command can be registered that it always initialized with some other command to run. The problem is how to provide a way to do it from console? What if the command that run other command not only run one but run multiple commands? How should I handle that? This lead to another problem too, that is how I create a command object. How can manager find out what it need to intialize a command object? That things must specify at console, and the manager shouldn't do anything with it. So, I need to provide a way to create/modify/delete a variable and track on that. I think I need to provide a default command to do this. All variable that create (I need to find a better name) from this command will be managed by some kind of container, maybe. I think a DI Container will fit this require cause I don't know exactly how should I manager the lifetime of contained object. Now the problem of this, I think, is how a request should be read. Somekind of regex should work here.

The problem I also find out is how should I manage variable that created from console? Should it be manage by console or by manager? The usage of it to provide instance for the command to run if needed. So what create the command instance? Manager. So should It is the manager will contain the list of variable? But a command don't know about the manager, how can it add a object to some kind of conatiner in manager in its execute method? If the ability to add variable to list is not something that not provide by the command but the manager, then I don't think there is any problem with it. I mean, it will go like how we register a new type of command, right? And this is also other problem. How we should register new command? The load should just load the file, I don't think it is good idea to make it have to register the command that maybe defined in the file too. Now it is other problem. Should I automatically register command or it need to tell that register the command. I currently on the side that make the user need to specify which command need to be register from console because I want all action with it like register, modify or delete should come from console. Maybe a command call register should able answer for this problem? But if the load just load dll or cs file, then it do not do anything with manager at all. So the register still have problem with manager. Also, in the load command, how can I manager file that being loaded? I must somehow manage all that file loaded, right? What if with the load command, I define it will execute with somekind of container that manage command too? I don't think its good idea to make the manager have to manage all file that loaded. It should be other type of container, and the load command should use it in its execute method. I think this can go the same way with register command. Its execute method will use a instance of type commandmanager as parameter. The question is how to provide these containers? I think it the main method of program. They will be default object that be initialized at the begin of program. And this make me wonder, should I have all object, include variable create at runtime and command that load and register in same container, or should them be splited based on it function? I don't know, and I don't think I will need to loop all of them to find something. Also, it will be easier to merge them if I need, so I don't think manage all object in only one container is good idea. No, I don't think so. What if I need to provide a parameter? I mean, a command and a variable can be a parameter for execute method, right? If they are in different container, it will be much harder for me to find the right one. So can they in the same container, the commands and the variables? With comands , I want to distinguish them with the request the respond. With the variables, obviously I want to distinguish them with their registered name. The command that called when run by manager, I don't think it should be something that be same list with the variables container. I mean, I want to distinguish them. The user can create their own intance of command for their own usage.

So, after all of this, what is the ability that a command manager need to provide? I don't think a manager should some how provide itself to a command. A command that call other command should be a command that be initialized with other command and call execute function of intialized command in its execute method. Command should be different with each other by the request or condition, or something like that. There should be a dictionary that take the condition as key to refer to a list of command that should be called when condition matched. Different type of manager will have different type of condition dictionary key. The command should be manager by some DI Container that can dynamic register. There are two core command: load and register. The load command just load code and the registerer register command to the manager. One thing more special is the load command should do its function on some kind of contianer that defined with its command, and that container will contain information about the file that this command has loaded. I want to go same way with the register command. The register command will register new executable command and new variable by work with two containers: one for commands and one for variables. That mean instead of one manager that allow contain and called command, we need something that somehow allow to receive the request, in this case is from console, check it with the command that be registered before to find the command that should be executed, then create an instance of it and call its execute method. If that command need something to run, that thing need to somehow get the instance that the command's execute method need from some container that can give us the instance we need. The question is, how? First, let call it as executer. We maybe have some abstract or interface that named executer, but think it later. I think it will need to intialize with a container that contain commands and variables. I don't know this is good idea or not, because thorugh time, these container will change dramaticly. But is it good idea to require porivde two container instance everytime the executer need to handle request? The container should be changed when it update outside, and for more secure, I will mark these container as something that cannot be modified. Now I will need a good name and consider if I need a family of executer or not. Now let move on to how this executer will find and call base on request. First at all, I think I will want some how all the instance registered with register command must have ICommand type, or at least they must have this kind of type. I don't want to adapt all the DI container just only to do that, so the type check maybe the best answer I can find. I don't know what different between a error happen at runtime and compile time. With this kind of error, the only difference I think is them just difference at the time user can know the error. If I try to make it found at compile time, it cost me huge effort to do that. So, I don't think its good idea to do that. The next question is,do I want the executor manage to start get request and handler it by itselft,or I want to setup it in the main function? What is different between them? Is it easier to defined behavior on main function? I don't know, so I think I will let the executor when received request just handle it. Now I found out another problem. I want the command when execute will respond something to console. But currently, I defined the command type return nothing in its execute method. So how exactly the command can respond after their work? First at all, specific type of executor may work with different kind of respond. The respond is not something I can define from beginning. The reason is what if I want the command define something that can do its own respond to console? In that situation, I think I have to provide a console so it can do it's function. Actually, I don’t know exactly how can I do that. Maybe that executor will take that respond and simply do its? Maybe that respond will have type be an action that take a console instance to do its work? I don't know exactly how it should be, but this is not time for this. I think it is posible to let the executor chose its own respond type.