# Purpose

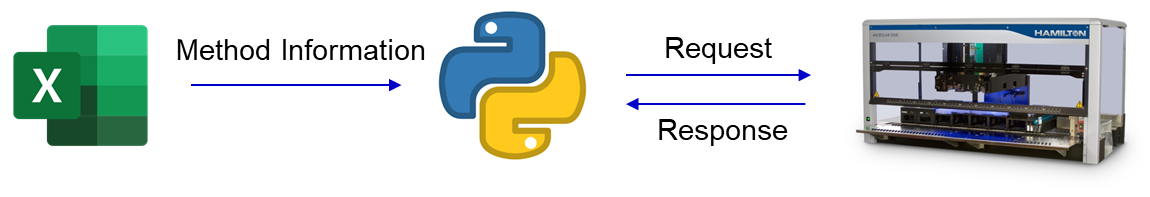
This protocol serves as the guidance to use the Automation Bare Necessities (ABN) Excel-based method editor on Hamilton automated liquid handlers. This document covers method creation and testing, includes information about each available Building Block, and details new instrument installation and configuration. This document should be updated as soon as new actions are made available to ABN users.

# Scope

ABN support is limited to Hamilton automated liquid handlers only. ABN will theoretically capture all programming and runtime errors before a sample run and prevent an error prone run from occurring. One important caveat is that ABN cannot capture all logic errors during method creation. If errors are experienced, it is important to report the errors as soon as possible so corrections can be implemented. This correction will prevent the error from accidently occurring during a run, which means that error reporting is the single most important responsibility for all ABN users.

# principle

ABN functions as a python-based method translator between Excel and Hamilton. During translation, ABN will request information from the Hamilton then use this as follow-up input information to create advanced functionality that is generally not possible on Hamilton systems.



ABN is a full-service software. This means that pipette tip selection, volume tracking, plate selection, reagent container selection, and deck loading is all performed automatically. This offers many advantages, but most importantly it means that the Hamilton is independent of its own configuration. This basically means that the Hamilton and ABN software, together, create a rock solid interface for fast method programming and robust method execution.

# Definitions

ABN – Automation Bare Necessities

SME – Subject Matter Expert(s)

Building Blocks (Blocks) –

Parent Plate – The plate block that comes immediately before all other blocks

# Method Creation

## Method Folders

### Storage Location

Methods are expected to be stored on the WIP drive at the following location:

\\amer.pfizer.com\pfizerfiles\Research\CHV\btxpharmsci\Instrument\ARD\\_Methods\_VisualMethodEditor

Within the above folder there are subfolders of each Method. A subfolder contains the following:

1. A template method named according to 5.1.3
2. An “Archive” folder to store previously run methods.

### Template

A Template method for starting a new method can be found at the storage location in 5.1.1 in the “Template” folder.

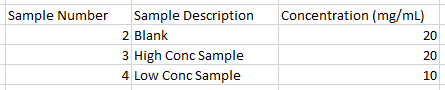
### Method Naming

New methods should be named as follows: “\_Template\_<Method Name>”. The “\_Template\_” designation informs users that the file is to be used as a starting file. Additionally, the template file should be a read only file. In a typical workflow, users will open the template file, fill out the worklist, and save as a new file. This new file is then used to run the Hamilton system.

## Excel Method Workbook Overview

### Worklist Sheet

The Worklist sheet contains information related to the samples. Once a method is created and validated, users who run the method will only modify this sheet. Thus, it is important to keep this sheet as simple as possible.



The worklist above is from a peptide mapping method. Only necessary information to run the method is present in the worklist.

### Method Sheet

The Method sheet is where you will design your method. This sheet is run by macros to ensure consistency across all methods in ABN.

### Solutions Sheet

The Solutions sheet is where method specific solution information is found. This sheet allows you to customize liquid handling and deck loading categories as needed.

### Other Sheets

As you test and validate your method you will notice a TestLog and RunLog sheet. Additionally, a PrepList sheet may appear. These sheets are automatically generated by the ABN python interface. Reading a Log sheet will be covered in depth in \_\_\_LOGS\_\_\_.

## Building Blocks

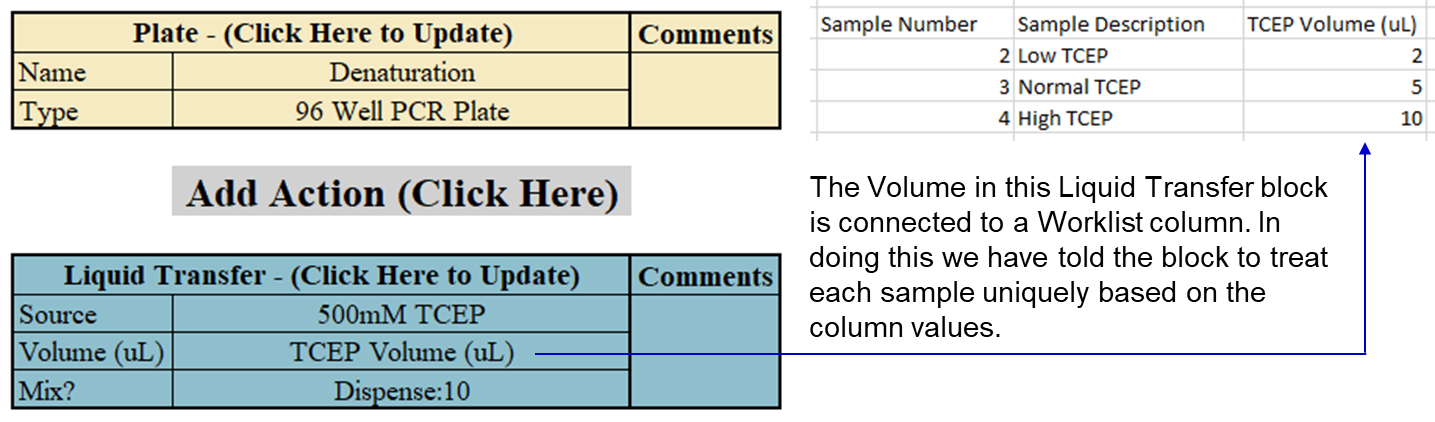
### High Level Description

Blocks is the term to describe all Pathways, Actions, and Modifiers. They are designed to convey information in a human readable sequence of events. Manual sample preparation can, similarly, be described as a linear sequence of events. Thus, blocks should theoretically be able to mimic all possible sample preparations in a lab setting. Blocks are made of 3 sections: Parameter Titles (left) which describe the information required; Parameters (middle) which is information provided by the user; and Comments (right) which is optional, but recommended, information to convey what is being accomplished to other users who may run your method.



### Worklist Interface

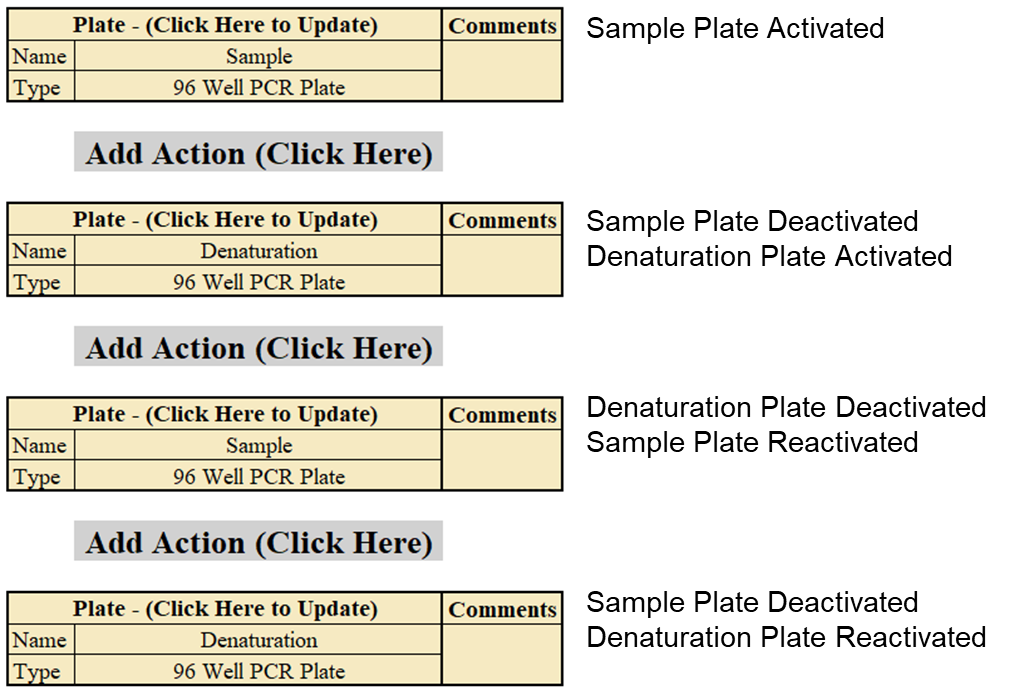
Blocks can interface directly to a sample worklist, which allows blocks to treat samples uniquely during processing. The primary use of the worklist interface is to enable development type workflows on automated systems. This goes against the thought process of automation where automation does a single method and does it consistently. ABN makes all the guarantees of a typical automation method but unlocks the full potential for development.



We can see above that each sample will receive the following volumes: Low TCEP -> 2uL, Normal TCEP -> 5uL, High TCEP -> 10uL. This functionality is not limited to volume because all parameters support the Worklist Interface. This will be confirmed in the Building Block Descriptions.

### Pathways

Pathways define a linear sequence of events. Pathways are defined with the Plate block only. The Plate block basically activates a new plate and deactivates the parent plate (the most recent plate). Other pathway options are available (Split Plate, Merge Plates, Finish), but behind the scenes they merely implement the Plate block in a different way. Plates can be Activated, deactivated, and reactivated as many times as you wish in a method without consequence.

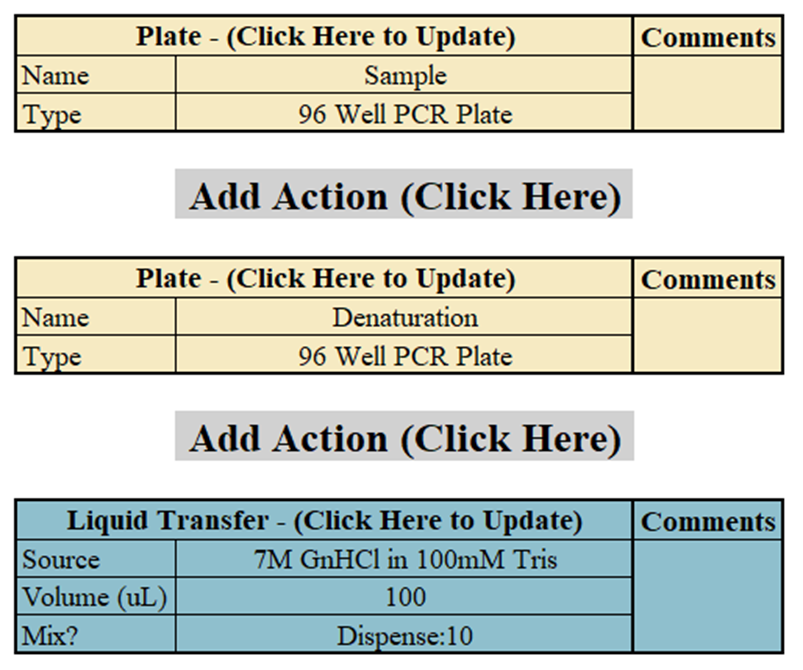


Now we need to understand the idea of a parent plate. The parent plate is the most recent Plate block in a pathway. Above, the first instance of Sample plate has no parent because it is the first plate in our linear timeline. Second, the parent plate of first instance of the Denaturation plate is the Sample plate. Third, the parent plate of the second instance of the Sample plate is the first instance of the Denaturation plate. So, who is the parent of the second instance of the Denaturation plate? The second instance of the Sample plate.

This brings us to context. Contextual information is very important because it tells the history of all blocks in the linear chain. The context of the first instance of Sample plate is nothing because it came from nothing. The context of the first instance of the Denaturation plate is “Sample”, because it is a direct descendant of that plate, and Sample has no context. Now it gets interesting… The context of the second instance of the Sample plate is “Sample:Denaturation”. And now we can start to see what is happening, we are creating a historical record of all plates in time. What is the context of the second instance of the denaturation plate? “Sample:Denaturation:Sample”. And if we added a new Plate block, the context would then be “Sample:Denaturation:Sample:Denaturation”. This information may not seem important, but this history is what allows us to split a plate into different pathways and will help you understand the impact of Modifiers.

### Actions

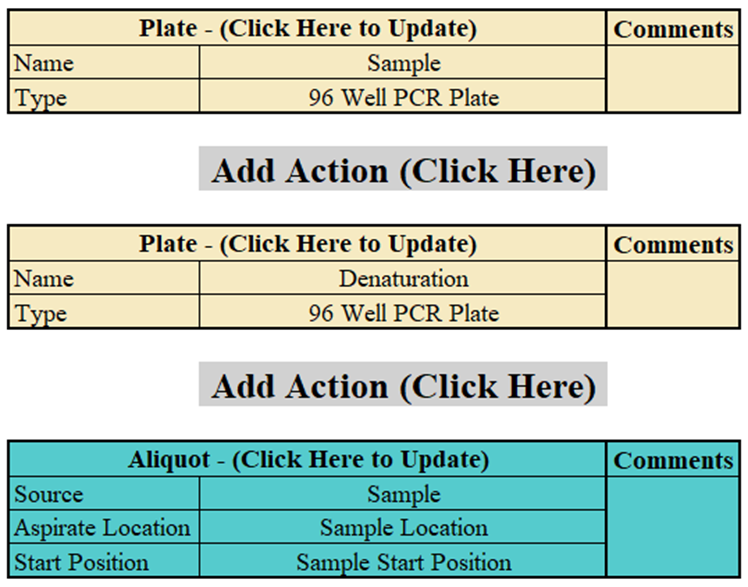
Actions act on plates that are defined in pathways. An action will always act on the Plate block immediately before it (most recent).



We can see above that Denaturation is the most recent Plate block. Thus, the Liquid Transfer block will dispense the liquid into the Denaturation plate. All following steps will continue to act on the Denaturation plate until a new Plate block is defined.

### Modifiers

Modifiers are used to significantly change the way Action blocks act in a Pathway. Modifiers are advanced blocks and, as such, the desired functionality should be confirmed in the log when creating a new method. Once desired functionality is confirmed, the method is suitable for use by all users.

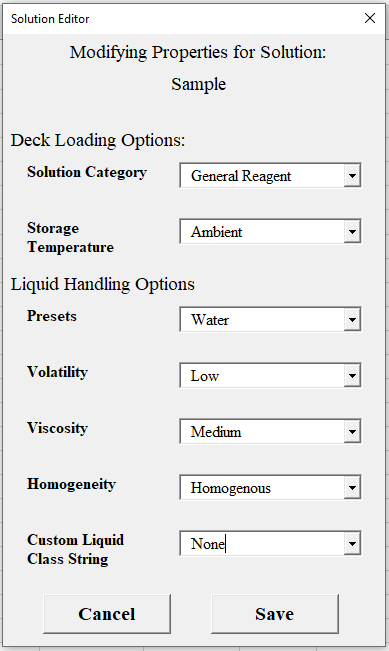


Above, we can see that Aliquot step is modifying the Sample plate and the Denaturation plate context. The underlying modification is not important, but it is significant. Now, when a Liquid Transfer or Dilute block appears, the pipetting style will be changed to reflect the information in the Sample Location worklist column.

## Solutions

### High Level Description

Proper solution handling is imperative for robust method execution on Hamilton systems. On both Hamilton and Tecan systems, liquid classes are used to optimize the liquid handling. Liquid classes allow users to adjust aspiration, dispense, and mix speeds, and, additionally, it is possible to define how quickly tips move after an aspiration step. This information is what prevents dripping of volatile solvents or incomplete aspiration of viscous solutions. The ABN interface abstracts away these parameters and, instead, attempts to boil liquid classes down to the simple inputs below.



### Deck Loading Options

Deck loading options are used to help the ABN interface load solutions and plates correctly on the deck. This can also provide consistency for users as the number of samples increase. In general, if nothing is changed in this section then ABN will always attempt to load solutions in the smallest possible container.

### Liquid Handling Options

Liquid handling options allow ABN to predict the liquid handling criteria. The ABN prediction will select the best fit liquid class and the minimum mixing requirements to ensure an error free liquid transfer. If the minimum criteria is exceeded by the method designer, then the higher criteria will take precedent. These options should be selected based on worst case scenario. There is no consequence for defining an aqueous solution as volatile or water as viscous. There is, however, huge consequence if you define Glycol as low viscosity. In most cases, these options will require testing to confirm liquid handling functions as expected. Note: Default solution parameters support low concentration aqueous solutions out of the box.

Plate liquid handling functions differently than reagent liquid handling. Liquid handling criteria for plates are calculated in real time. The calculation considers the liquid composition in each well individually. It is not possible to override this functionality as plate solutions are constantly changing. One caveat: if liquid has never dispensed into a plate then the plate will default to the user entered liquid handling options.

## Building the Method

### The Thought Process

When building a method, it can be confusing to translate a manual workflow into an abstract automated workflow. To combat this, ABN provides blocks that distill complex liquid handling actions into simple manual steps. These manual steps are meant to mimic a lab based workflow as closely as possible. Thus, the best way to approach this problem is to do the following: Consider a plate as a container (tube) and yourself as the Hamilton. Now each block can be imagined as an action you perform manually in the lab.



In the example above, we can see one method to translate SOI text into a method. For demonstration purposes this method has not been scaled up or down, which is justified in 5.2.2. So, we are going to read the method, imagine ourselves performing the protocol in the lab, then select the correct block to accomplish each physical task. First, we can see there are two “containers” being used: A sample container, and a reduction container. So, we will add a plate where the sample is expected to be located (Sample) and a plate where we will perform the method (Reduction). Then, the method requests that we dilute a sample solution to 10mg/mL in 30uL, and the obvious best block for this is the Dilute block. We may not know the **Starting Concentration (mg/mL)** ahead of time, so we can create a worklist column where to user will enter the information. Next, we are transferring 2 solutions to our diluted sample: 257uL of 7M GnHCl in 100mM Tris pH 8.2 and 3uL of 0.5M TCEP, and the best block for this is two Liquid Transfer blocks. Finally, we are instructed to incubate for 55 minutes at 40C, so we will use an Incubate block.

### Translating an SOI Effectively

In section 5.5.1 we showed a simple example to translate an SOI into an automated workflow. In most cases, however, it will be necessary to scale volume to better suit a Hamilton system.

For example, ABN Hamilton systems typically have plates that support 200uL, 400uL, 1200uL, and 2000uL. So, it would not be wise to have a total volume of 450uL in a 1200uL plate. Instead, we will scale the volumes, either up or down, to ensure that the volume is suitable for a given plate size. ABN will, of course, select the best plate for a given method, but it is important to design around a specific plate size. This ensures the method will be robust for all users.

As another example, methods are possible where the final volume is 1200uL, but there is a 100uL incubation before the final dilution to 1200uL. In this case we would want to use two plates. We would have one small plate for the 100uL incubation (**Name**: “Low Vol Incubation”) and one large plate for the dilution (**Name**: “High Vol dilution”). Then after the incubation we can transfer the 100uL from the small plate into the larger plate for final dilution.

As a method designer it is important to make these considerations to ensure method robustness. This is true if you are designing a method in ABN or in the Hamilton programming software. ABN is merely an interface to simplify method creation and execution for all users. It cannot guard against poor method design.

## Method Testing and Validation

### Types of Tests

There are two types of tests in ABN: Programmatic, and Physical. ABN software is designed to expedite method delivery time. As such, in almost all cases a programmatic test is sufficient when writing a method.

A programmatic test is when ABN runs the software before method execution on a Hamilton. Programmatic testing can capture both logic and runtime errors. Additionally, this testing type will generate a TestLog sheet in the excel workbook where the method creator can read and confirm steps are occurring as intended.

A physical test is when the method creator runs the method on an actual Hamilton system. Physical testing is typically only required during the following: high temperature incubations (95C), liquid transfers on extremely volatile solvents, and if highly variable blocks such as Magnetic Beads or Vacuum are included.

### What to look for

As a method creator it is imperative to confirm the correct sequence of events in a TestLog sheet. This is particularly important if modifiers are used in your method. Additionally, if you add new blocks to an already established method you must confirm the new additions in the TestLog. Once a method is confirmed there is no reason to check the TestLog in the future.

## Log Sheets

### Log Contents

### How to Read the Log

# Building block Descriptions

## Notes

### Building Blocks Availability

Not all blocks may be available on your system. Available blocks are configuration dependent. If you would like a block implemented, please reach out to a Hamilton SME to discuss feasibility.

### Building Block Changes

Blocks are subject to change (this means color as well). You should be notified before changes are pushed to your system. If you notice a block has changed and do not understand the change, you should reference this document. If information has not been updated for that block. Please reach out to a SME immediately.

### Section Goals

Input options which are configuration dependent and will not be discussed here. Instead, this section will cover the meaning behind each input and potential block limitations as required.

## Pathways

### Plate



**Block Description:** This block creates a new pathway for blocks to be executed. All blocks that follow this step will act on a virtual plate with the **Name** argument.

**Name:** Can be any text. This names the new pathway you are creating.

**Type:** Can be any value from the dropdown. This argument controls what type of labware can be used for deck loading. This does not limit the volume of the labware.

### Split Plate



**Block Description:** This block splits a pathway into two new pathways. These pathways are processed together, meaning no delay is expected for actions across pathways.

**Plate Choice:** This chooses the path for each sample. There are always 5 available options: Split, Concurrent, the 2 names you choose for the new pathways or a Worklist Column. The **Split** choice means that the sample will be processed on both pathways, but the volume will be halved. The **Concurrent** choice means that the sample will be processed on both pathways, but the volume will not be halved. If you choose the name for **Pathway 1** or **Pathway 2**, that means that the sample will be processed on only that pathway and the volume will not be halved.

**Plate Name 1:** Can be any text. This names the new pathway you are creating.

**Plate Name 2:** Can be any text. This names the new pathway you are creating.

### Merge Plates



**Block Description:** This block merges 2 pathways into a single pathway. A merge block is required on each pathway you would like to merge.

**Plate Name:** Is the name of the parent plate on the other merging pathway.

**Continue Here?:** The available options are always **Yes** or **No**. If both Merge blocks are Yes, then this step acts only as a synchronization tool. If one of two blocks are Yes, then this will merge both pathways together. If the volumes were halved by a Split Plate block then they will stay halved once merged.

### Finish



**Block Description:** This block terminates flow on a pathway. This block should always be the last block in a pathway.

## Actions

### Dilute



**Block Description:** This block performs a dilution from a **Source** and uses **Diluent** to dilute the **Source** from the **Starting Concentration (mg/mL)** to the **Target Concentration (mg/mL)**. Behind the scenes, this block is made from two Liquid Transfer blocks. This block will always pipette the largest volume first, which ensures complete transfer down to 1uL. The formula C1V1 = C2V2 is used to determine the volume of **Source** and **Diluent**.

**Source:** Can be any text, plate name, or a worklist column

**Diluent:** Can be any text, plate name, or a worklist column

**Starting Concentration (mg/mL):** Can be any number or a worklist column

**Target Concentration (mg/mL):** Can be any number or a worklist column

**Target Volume (uL):** Can be any number or a worklist column

**Max Source Volume (uL):** Can be any number or a worklist column. This parameter limits how much volume can be taken from the **Source**. Example: If the dilution requires 100uL of **Source** but this parameter is 80uL, then only 80uL will be taken from **Source**. The remaining 20uL will be taken from **Diluent.**

### IMCS SizeX Desalt



**Block Description:** This block uses IMCS Size Exclusion (SizeX) tips to desalt intact protein sample. Sample must be denatured, reduced, and alkylated prior to desalting. Only SizeX100 is supported.

**Source Plate:** Can be any plate name

**Waste Plate:** Can be any plate name

**Equilibration Buffer:** Can be any text or any plate name

**Load Volume:** This is the load volume. The typical yield for IMCS tips is 80% of the **Load Volume**. It is important to consider this in following Liquid Transfer or Dilute blocks. Additionally, the load concentration for IMCS tips should be 1mg/mL if possible.

**Elution Method:** The IMCS tips are not as robust as the Nap5 columns. As a work around, various **Elution Methods** are available to ensure the desalted sample meets your analytical needs.

### Incubate



**Block Description:** This block can heat, cool, or perform an on deck ambient incubation.

**Temp (C):** Can be Ambient or a number.

**Time (min):** Can be a number.

**Shake (rpm):** Can be a number.

### Liquid Transfer



**Block Description:** This block performs a simple liquid transfer from **Source** into the parent plate.

**Source:** Can be text, a plate name, or a worklist column

**Volume (uL):** Can be a number or a worklist column

**Mix?:** This option gives users the ability to select to mix before aspiration, after dispensing, or both. Additionally, the user can choose the number of mixing cycles. A single mixing cycle is both an aspirate and dispense event.

### Magnetic Beads



**Block Description:** This block condenses beads according to the **Hold Time (min)** parameter, removes the liquid present in the wells, adds the desired **Storage Buffer Volume(uL) of Storage Buffer**, then resuspends the beads in the **Storage Buffer** via pipette mixing. The workflow describe constitutes a single **Repetition** and will be repeated **Repetition** times.

**Magnetic Beads Plate:** Can be a plate name

**Storage Buffer:** Can be text, a plate name, or a worklist column

**Storage Buffer Volume (uL):** Can be a number or a worklist column

**Hold Time (min):** Can be a number

**Repetitions:** Can be a number

### Notify



**Block Description:** This block will email and/or text a user a given **Subject** and **Message**.

**Wait On User:** If Yes, then this block will wait for the user to respond before proceeding. This wait will halt all execution on the Hamilton. This includes execution in other pathways. If No, the user will only be notified.

**Subject:** Can be text.

**Message:** Can be text.

### Vacuum



**Block Description:** This block will, in order, move the parent plate into the vacuum manifold, move the desired **Vacuum Plate** on top, add the **Source** to the **Vacuum Plate**, wait for **Pre Vacuum Wait (min)**, apply a vacuum according to **Pressure Difference (mtorr)** for **Vacuum Time (min)**, then remove the Vacuum plate and collection plate from the manifold.

**Source:** Can be text, a plate name, or a worklist column

**Volume (uL):** Can be a number or a worklist column

**Vacuum Plate:** This plate is dependent on your Hamilton configuration. **Vacuum Plates** require significant development across Hamilton systems so are not immediately transferrable.

**Pre Vacuum Wait (min):** Can be a number

**Pressure Difference (mtorr):** Can be a number

**Vacuum Time (min):** Can be a number

## Modifiers

### Aliquot



**Block Description:** This block modifies the **Source** and parent plate sample knowledge to enable aliquoting. This means that if you perform a Liquid Transfer or Dilute block after this step, the sample location itself will be modified. This allows users to load sample in a single well of the **Source** and aliquot into any number of different wells in the destination.

**Source:** Can be a plate name

**Aspirate Location:** Can be a worklist column

**Start Position:** This specifies how the sample knowledge is changed. Sample Start Position means that the **Source** is expected to be loaded at the chosen sample start location. Plate Start Position means that the **Source** is expected to be loaded in the first well.

### Pool



**Block Description:** This block modifies the **Source** and parent plate sample knowledge to enable pooling. This means that if you perform a Liquid Transfer or Dilute block after this step, the sample location itself will be modified. This allows users to take sample from any number of different wells in the **Source** and pool into a single well in the destination.

**Source:** Can be a plate name

**Dispense Location:** Can be a worklist column

**Start Position:** This specifies how the sample knowledge is changed. Sample Start Position means that the **Source** is expected to be loaded at the chosen sample start location. Plate Start Position means that the **Source** is expected to be loaded in the first well.

### Preload Liquid



**Block Description:** This block allows the user to inform the Hamilton that the given **Volume (uL)** is already present in the wells. This is important because well volume is tracked throughout the entire method execution.

**Volume (uL):** Can be a number or a worklist column

# Run a method

Work in Progress

# Troubleshooting

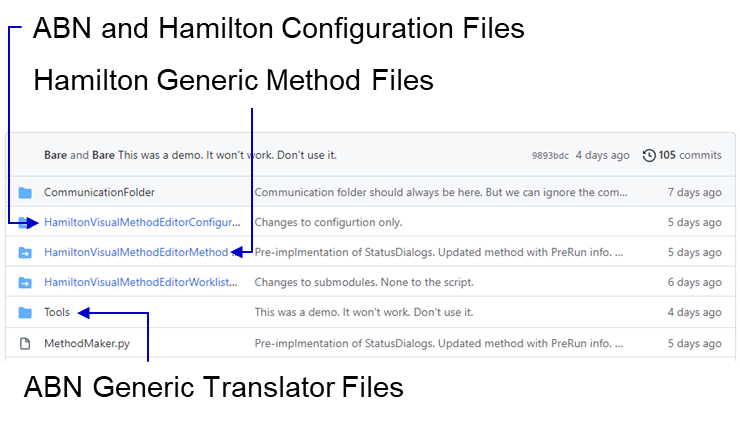
Do I really need this?

# Advanced Information

## Behind the Scenes

### Folder Structure

ABN contains 3 important folders: Tools, HamiltonVisualMethodEditorConfiguration, and HamiltonVisualMethodEditorMethod. The HamiltonVisualMethodEditorConfiguration folder is used to create a custom configuration for your Hamilton system. The HamiltonVisualMethodEditorMethod folder houses the generic method that ABN uses to communicate and access modules. Tools are all ABN files used to perform translation from excel to Hamilton commands.



### Why Blocks?

Blocks are easy to visualize and contain information in a pleasant way. Additionally, a blocks convey information in a stepwise fashion. All lab protocols are stepwise, so blocks are the simplest and most recognizable object to represent a workflow.

Blocks also simplify method creation. A dilution step is about 100 lines of Hamilton code. By creating a block, we can re-use the 100 lines of Hamilton code as many times as needed to accomplish our method goal. But the method size stays compact and human readable.

### How does ABN Translate Blocks?

Block translation occurs completely in python by the block designer. Each block contains various Hamilton commands that occur behind the scenes. For example, an Incubate block is made up of the following steps. An initial heating step to prepare the heater for the incubation, which happens as early as possible. A confirmation step to ensure the heater is at or close to temp. A plate transfer step that moves the plate and lid from the deck to the heater. A timer step that waits for the incubation to complete. And finally, a plate transfer step that moves the lid and plate from the heater back to the deck. The preceding physical actions are implemented as Hamilton modules. The Incubate step uses the Transport, Heater, and Lid modules.

### Why a Command Response System?

Code separation is very important in ABN. The goal of ABN is to leverage Hamilton libraries (Modules) to create functionality. The beauty of this system is that all Hamilton modules can be used without ABN, promoting a re-usable code atmosphere.

ABN doesn’t know anything about the Hamilton and the Hamilton doesn’t know anything about ABN. So, ABN must request the information it needs to complete a task. Simply, a command response system promotes code separation but allows for information communication, which theoretically allows ABN to run on Tecan systems as well without any changes.

This also mean that new blocks are easier to implement. The typical workflow for block creation is as follows. 1. Create a proof of concept library that proves feasibility, 2. Generalize the library as a module, 3. Design and connect the block to the Hamilton modules. Done.

### Automated Container Selection and Deck Loading

I don’t want to do this…

## ABN Architecture

### Input and Output (IO)

### User Blocks

### Hamilton Commands

## Hamilton Architecture

### The Generic Method

### Harmonized Deck Layout

### Configuration Dependent Modules

### The Deck Loading Library

## ABN Software Installation (Local, Hamilton)

### Critical Software

**Python**

**Git**

**Visual Studio Code**

### Create Folders

### Clone from Github

### Import the Hamilton Method

## ABN and Hamilton Configuration