**ATRYBUTY**

Atrybuty są modyfikatorami dla parametrów i umożliwiają zmianę lub definiowanie ich zachowania.

**PARAMETER**

Parametry mogą być także opisywane za pomocą słowa kluczowego Parameter. Jego użycie pozwala na dostęp do dodatkowych funkcjonalności. Nie jest konieczne, ale jest zalecane w zaawansowanych funkcjach. Parameter przyjmuje opcjonalne argumenty, każdy oddzielony przecinkiem. Argumenty te zmieniają zachowanie funkcji w stosunku do parametru, który jest opisywany.

Mandatory - Przykładowo Parameter może przyjmować argument Mandatory, który określa, że podanie wartości dla tego parametru przy wywoływaniu funkcji jest wymagane

Position – określa którą wartość w kolejności ma przyjąć parametr z przekazanych nienazwanych parametrów. Domyślnie nienazwane parametry są przypisywane w kolejności ich zdefiniowania. Ten argument pozwala na zmianę tej kolejności bez zmiany definicji.

ParameterSetName – definiuje zestawy parametrów i zezwala funkcji na wykorzystywanie ich w zależności od potrzeb. Jeśli użyjemy parametrów zdefiniowanych w jednym parametersetname, to te z innego nie będą dostępne.

ValueFromPipeline – definiuje, że wartość parametru może być przyjmowana z pipeline’u

ValueFromPipelineByPropertyName – definiuje, że wartość parametru może być przyjmowana z pipeline’u jeśli nazwa obiektu ma taką samą nazwę jak parametr

ValueFromRemainingArguments – definiuje, że parametr przyjmuje wszystkie wartości, które nie zostały przekazane do innych parametrów

HelpMessage – wyświetla pomoc dla wymaganego parametru, jeśli nie została podana jego wartość

**ALIAS**

Ustala alternatywną nazwę dla parametru. Ilość aliasów możliwa do przypisania do parametru nie jest limitowana.

Kolejne parametry będą umożliwiały sprawdzanie czy wartość parametru ma dozwoloną wartość.

**AllowNull**

Pozwala na przekazanie wartościu $null do wymaganego parametru.

**AllowEmptyString**

Pozwala na przekazanie pustego stringa do wymaganego parametru o typie string

**AllowEmptyCollection**

Pozwala na przekazanie pustej kolekcji do wymaganego parametru

**ValidateCount**

Określa minimalną i maksymalną ilość elementów przekazywanych do parametru.

**ValidateLength**

Określa minimalną i maksymalną ilość znaków w przekazanym parametrze.

**ValidatePattern**

Określa wyrażenie regularne, które jest porównywane z wartością przekazywanego parametru.

**ValidateRange**

Określa zakres liczbowy w którym powinna się znaleźć wartość parametru.

**ValidateScript**

Określa skrypt, który jest używany do zweryfikowania wartości parametru.

**ValidateSet**

Określa listę elementów możliwych do przypisania jako wartości parametru.

**ValidateNotNull**

Określa, że wartość parametru nie może być $null.

WAŻNE: Dokumentacja mówi, że jeśli typ parametru to string to przekazanie do niego $null spowoduje automatyczne przekonwertowanie tego $nulla na pusty string i walidacja się powiedzie.

**ValidateNotNullOrEmpty**

Określa, że wartość parametru nie może być $null oraz nie może być pusta

**ValidateDrive**

Określa psdrive’y, które mogą być użyte w ścieżce. Nie sprawdza czy plik/folder istnieje, weryfikuje tylko drive.

**ValidateUserDrive**

Określa, że tylko user drive może być użyty w ścieżce. Używane przy JEA.

**Dynamic parameters**

Dynamiczne parametry są dostępne w funkcji tylko jeśli wystąpią określone warunki np. została podana określona wartość innego parametru.

**Switch**

Jest to parametr, który nie przyjmuje wartości. Użycie parametru powoduje ustawienie wartości zmiennej, która go reprezentuje na $true. W przeciwnym przypadku zmienna otrzymuje wartość $false.

**CMDLETBINDING**

Atrybut CMDLETBINDING powoduje, że funkcja zachowuje się analogicznie do skompilowanego cmdleta, dzięki czemu otrzymujemy dostęp do wszystkich jego zalet.

Użycie tego atrybutu powoduje, że otrzymujemy dostęp do automatycznej zmiennej $PSCmdlet ale tracimy $args.

Cmdletbinding może przyjmować poniższe argumenty, które modyfikują przepływ funkcji lub dają dodatkowe możliwości.

**SupportsShouldProcess**

Argument dodaje parametry Confirm i WhatIf do funkcji. Parametr confirm prosi użytkownika o potwierdzenie przed wykonaniem każdego elementu z pipelineu. WhatIf informuje o akcji, która zostanie wykonana bez jej wykonywania.

**ConfirmImpact**

Definiuje czy wywołanie funkcji ma być potwierdzane poprzez odwołanie do metody ShouldProcess

**DefaultParameterSetName**

Definiuje który zestaw parametrów ma być użyty, jeśli nie można określić który użyć.

**HelpURI**

Definiuje adres url do pomocy online dla funkcji.

**SupportsPaging**

Argument dodaje parametry First, Skip I IncludeTotalCount służące do kontrolowania zwracanych elementów z funkcji.

* First: Zwraca tylko n elementów
* Skip: Pomija pierwsze n elementów
* IncludeTotalCount: Zwraca ilość obiektów

**PositionalBinding**

Definiuje czy przypisywanie nienazwanych parametrów ma być realizowane.

**OutputType**

Atrybut definiuje listę typów .Net, które są zwracane przez funkcję. Jest to tylko informacja dokumentująca funkcję i nie wpływa na jej działanie. Zdefiniowanie jednego typu i zwracanie innego nie wpływa na działanie funkcji.