**TimeWorksPlus Scripting**

**Processing Rules** The Processing Rules options below may be used to add additional fields to Employee Setup,

or additional columns to the time card. These may then be used in scripting.

(**X**) Accruals

AccrualFactor Adds an Accrual Factor field to Employee Setup, accessible as

"employee.accrualfactor" in scripts.

(**X**) Additional user fields

Birthday Enables the user to add a birthday field to Employee Setup.

Departments Enables the user to add extra department fields to Employee Setup.

HomeFields Enables the user to add extra home fields to Employee Setup.

Locations Enables the user to add extra location fields to Employee Setup.

(**X**) Scripting

ShiftScript This rule adds a Shift column to time cards, and runs a custom script to

determine the shift.

**User Defined Variable Names:**

May contain UPPER and lower case letters **A** - **Z** and **a** – **z** Examples: HoursWorked

May contain numerals **0** – **9** Week52

May contain underlines **\_** Worker\_Age

Must begin with either a letter or a **$** $TimeOff

Variables which begin with a letter are global for all subsequent Script Boxes Minimum\_Wage

Variables which begin with a $ are local to only the current Script Box $PieceRate

The variable type is determined based by the first assignment statement:

Examples: HoursWorked = 10; HoursWorked is a numeric variable type

HomeDepartment = “Office” HomeDepartment is a string variable type

Lunch\_Time = OutTime Lunch\_Time is a time variable type

$PunchDate = reportingdate.date $PunchDate is a date variable type

$PunchDateTime = indt $PunchDateTime is a datetime variable type

$PtoEligible = True PtoEligible is a Boolean (logical) variable type

**Numeric Operators:**

+ Add Numeric Example: Variable1 = Variable2 + Variable3 3 = 1 + 2

- Subtract Numeric Example: Variable1 = Variable2 - Variable3 1 = 3 – 2

\* Multiply Numeric Example: Variable1 = Variable2 \* Variable3 6 = 2 \* 3

/ Divide Numeric Example: Variable1 = Variable2 / Variable3 2 = 6 / 3

% Remainder (or Mod) Example : Variable1 = Variable2 % Variable3 2 = 6 % 4

**String Operators:**

+ Concatenate strings Example: String1 = String2 + String3 “123” = “1” + “23”

**Logical Operators:**

and And operator Example: if ((Variable1 = 1) and (Variable2 = 2))

or Or operator Example: if ((Variable1 = 1) or (Variable2 = 2))

= is equal Example: if (Variable1 = Variable2) Returns true/false

== is equal Example: if (Variable1 == Variable2) Returns true/false

<> Not equal Example : if (Variable1 <> Variable2) Returns true/false

!= Not equal Example: if (Variable1 != Variable2) Returns true/false

< Less than Example : if (Variable1 < Variable2) Returns true/false

<= Less than or equal to Example: if (Variable1 <= Variable2) Returns true/false

> Greater than Example : if (Variable1 > Variable2) Returns true/false

>= Greater than or equal to Example: if (Variable1 >= Variable2) Returns true/false

Contains operator Example: if(String1 Contians String2) Returns true/false

Startswith operator Example: if(String1 Startswith String2) Returns true/false

Endswith operator Example: if(String1 Endswith String2) Returns true/false

**End of Statement Operator:**

**;** Ends each statement Example: Payrate = 10**;**

**Block and Expression Grouping Operators**

{} {block of statements}

() (expression)

**Comment Operators:**

// Anything following the // on a line will be ignored

Examples: // This is a comment line

If (category = "Regular") // check on category

/\* Begin comment block

\*/ End comment block Example: /\*

if (Category = "Regular") <--- this line will be ignored

{ <--- this line will be ignored

Payrate = 10; <--- this line will be ignored

} <--- this line will be ignored

\*/

**Decision Operators:**

If (Logical Expression) { Example 1: If (Department = "1") {

Statements; Payrate = 12;

} }

If (Logical Expression) { Example 2: If (Category = "Regular"){

Statements; $RegHours = Hours;

} Else { } Else {

Statements; $RegHours = 0;

} }

If (Logical Expression) { Example 3: If (Employee.YearsOfService >= 10) {

Statements; $AccrualHours = 5;

} Else If (Logical Expression) { } Else If (Employee.YearsOfService >= 5) {

Statements; $AccrualHours = 3.33;

} Else If (Logical Expression) { } Else If (Employee.YearsOfService >= 0) {

Statements; $AccrualHours = 1.67;

} Else { } Else {

Statements; $AccrualHours = 0;

} }

**General Functions:**

Val(string) Returns a number from of a string

Example: Val("5.555")) = 5.555

Abs(number) Returns a positive number

Example: Abs(-5)) = 5

Cint(number) Returns the truncated interger value of the NumericVariable

Example: Cint(1.2) = 1

Cint(2.0) = 2

Round(value,decimal) Returns a number rounded to decimal places

(All script Boxes except Rounding) Round(12.3456,2) Returns 12.34

Left(string, number) Returns the first number of characters in the string of characters

Example: Left("this is the best",4)) = "this"

Right(string, number) Returns the last number of characters in a string of characters

Example: Right("this is the best",4)) = "best"

Mid(string, begin, number) Returns the number of characters starting at Begin, from the string

Example: Mid("this is the best",6,2)) "is"

Cstr(number) Returns string of Numeric Variable

Example: Cstr(1.2) = “1.2”

**Script Box Processing Order:**

1. Rounding
2. Script
3. Split
4. ReportingDate
5. AddEntry
6. SplitPostReportingDate
7. Shift
8. PayRate
9. Accruals
10. Overtime

The following are keywords in the scripting engine of **TimeWorksPlus**. Most are able to be used in all scripting boxes, which can be accessed through Processing Rules. But there are some that will only work in certain script boxes.

**Global - Applicable to all Script Boxes**

True True constant

False False constant

**Employee.** Accessed through **Employee.<variable>**

.YearsOfService Returns a decimal number

.DaysOfService Returns a decimal number)

.Anniversary Returns a whole number on the anniversary date of the employee

.Monthiversary Returns a whole number on the anniversary date of the employee

.Title

.FirstName

.LastName

.Code

.Designation

.StartDate

.EndDate

.Department

.Location

.Supervisor

.Home1

.Home2

.Home3

.AutoLunchHours

.LunchMinutes

**Defined Variables:**

Hours hours worked

Minutes minutes worked

Seconds seconds worked

BreakSeconds break time in seconds

MinutesOut amount of time elapsed between the current in punch to the previous out punch

MinuteStil amount of time elapsed between the current out punch to the next in punch

PunchSet the is times punch set number

Category the category of the punch

PunchDate date of the punch

InDt In DateTime - Formated: YYYY-DD-MM HH:MM:SS

OutDt Out DateTime - Formated: YYYY-DD-MM HH:MM:SS

InTime the time of the clock in punch

OutTime the time of the clock out punch

InIsPresent true if the InTime is present

OutIsPresent true if the OutTime is present

InIsMissing false if the InTime is not missing

OutIsMissing false if the OutTime is not missing

IsTimes true if there are times

IsHours true if only hours

IsPayOnly true if dollars only

InIsEdited true if InTime was edited

OutIsEdited true if OutTime was edited

HoursToPunch hours previous to the punch in question on that day

HoursToPunchOt hours previous to the punch in question on that day that are overtime eligible

LineToNow the amount of seconds from the last punch to what it is right now

InIp the IP address of the in punch

OutIp the IP address of the out punch

**Defined Functions:**

Within(Find,”Item|Item|Item|…”) Returns true or false

Example: Within( department , "100|110|200|210" )

Translate(Item,”Value1| Value2| Value3|…”, “Out1|Out2|Out3|…|Defualt”) Changes the selected Item from Value1 to Out1

Example: Translate( department , "100|110|200|210", “Dock,Shipping,Accounting,Billing,Warehouse” )

DateAdd (interval, number, DateTime) Returns a Date or Time or a DateTime

Interval can be one of the following:

Returns a Time or a DateTime:

“s” Seconds

“m” or “n” Minutes

“h” Hours

Returns a Date or a DateTime:

“d” Days

“mm” Months

“y” or “yyyy” Years

Examples:

Returns a Time or a DateTime

DateAdd("s", 10, InTime) Returns: Intime + 10 seconds

DateAdd("m", 30, InTime) Returns: Intime + 30 minutes

DateAdd("h", 2, OutTime) Returns: OutTime + 2 hours

Returns a date:

DateAdd("d", 7, PunchDate) Returns: PunchDate + 7 days

DateAdd("mm", 3, InDt) Returns: InDt + 3 months

DateAdd("y", 3, OutDt) Returns: OutDt + 3 Years

DateSerial(Year,Month,Day) Returns a date Example: DateSerial(reportingdate.year,12,31))

Weekday(date) Example: Weekday(punchdate))

Returns: M- Monday  
T- Tuesday  
W- Wednesday  
R- Thursday  
F- Friday  
S- Saturday  
U- Sunday

AddAlert(“Message”) adds an alert message to the time card

Example: AddAlert("Late"))

Unpay(hours) sets the amount defined in hours to unpaid.

Example: Unpay(hours))

Touches(time1, time2, time3,…) Returns true/false if punch set touches (ie: is within) specified times

Example: Touches(6:00am,3:00pm))

Overlaps(time1, time2, time3,…) Returns true or false if punch set overlaps specified time

Day(date) Returns the day of the date object

Month(date) Returns the month of the date object

Year(date) Returns the year of the date object

IsEdited(Property) Returns true if the string property given was edited for the punch

Property can be: Category  
Payrate  
FieldName of any clock prompt

ToSeconds(Time) Converts time to Seconds

Example: $InMins = ToSeconds(InTime)

**Specific - Applicable to only certain Script Boxes**

Script Box: **Rounding**

Round(Expression) Rounds all punches

Example: Round (“9:00am – 9:30am = 9:00am”)

Rounds time between 9:00am and 9:30am to be 9:00am

Example: Round (N15)

Rounds to the nearest 15 minutes

RoundIn(Expression) Rounds all **In** punches

RoundOut(Expression) Rounds all **Out** punches

RoundEnds(Expression) Rounds the first **In** and the last **Out** punch of each day (roundends("C15");)

The basic parameters are:

N - NEAREST

U – UP

D – DOWN

C - Company (Rounds in favor of the company, i.e. INs UP and OUTs DOWN)

You can qualify the above parameters with:

I – IN punches only

O – OUT punches only

Examples:

Round ("N10") Rounds to the NEAREST 10 minutes interval.

RoundEnds ("C10") Rounds in favor of the company by 10 minute intervals.

RoundIn ("U15") Rounds the IN punches UP (or "ahead”) to the next 15 minute interval.

RoundOut ("U5") Rounds the OUT punches UP to the next 5 minute interval.

Round ("ID1") Rounds the IN punch DOWN (or "back”) to previous 1 minute interval.

Round ("OD15") Rounds the OUT punch DOWN to the previous 15 minute interval.

RoundToSchedule(a1,a2,b1,b2) Rounds to the schedule found in Employee Setup

a1 Number of minutes prior to shift that we are rounding to the scheduled start time (if they are early)  
a2 Number of minutes after the start of the shift that we round back to the start time (if they are late)  
b1 Number of minutes after the end of shift that we are rounding to the end time (if they leave late)  
b2 Number of minutes prior to the end of the shift that we are rounding to end time (if they leave early)

Script Box: **Script**

(Nothing specific)

Script Box: **Split**

Split(Time, HideSplitTime) Takes up to two parameters in the Split script box

Time is required

HideSplitTime is optional, and is either “True” or “False”, depending if you want to hide

the time at which you are splitting the punch. This is used when you want

to generate a new punch, and split off the last second of the punch.

Examples: Split(12:00am)

Split(InTime,False)

Script Box: **ReportingDate**

Tomorrow(0) Pushes the punch to the next day Example: Tomorrow(0)

Yesterday(0) Pushes the punch to the previous day Example: Yesterday(0)

Script Box: **AddEntry**

AddEntry(type, amount, category) type has to be either “hours” or “amount”

Adds an entry to the time card

Script Box: **SplitPostReportingDate**

Split(Time, PushToTomorrow, HideSplitTime) Takes up to three parameters in the SplitPostReportingDate script box

Time is required

PushToTomorrow is optional, and is either “True” or “False. Is used in

splitPostReportingDate, because the Reportingdate of the punch is already

set, so this can choose whether or not to push the punch to the next day. This would be used if you wanted to split at midnight on a holiday, and you want to push the punch to the next day.

HideSplitTime is optional, and is either “True” or “False”, depending if you want to hide

the time at which you are splitting the punch. This is used when you want

to generate a new punch, and split off the last second of the punch.

Examples: Split(12:00am)

Split(InTime,True,False)

Script Box: **Shift**

Shift Refers to a new column on the time card created by this scripting processing rule.

Script Box: **Payrate**

PayRate Pay Rate

IsFirstToday Is this the first punch for Today

IsLastToday Is this the last punch for Today

ReportingDate. ***(Can also be used in all scripting boxes after the ReportingDateScript)***

.Date Returns the Date

.Todaysdate Returns the Date of today (real time)

.Year Returns the Year

.Month Returns the Month (1-12)

.Weekday Returns the weekday (1-7)

.WorkWeekStart Returns Beginning date of the week

.WorkWeekEnd Returns End date of the week

.IsFirstDayOfMonth is this the first day of the month (true/false)

.IsLastDayOfMonth is this the last day of the month (true/false)

.PayPeriodStart Returns the pay period start date

.PayPeriodEnd Returns the pay period end date

.IsHoliday([Date]) Is this a holiday (true/false) Date is optional

Example: ReportingDate.IsHoliday(DateSerial(ReportingDate.Year, 12, 25))

.IsLastPunchPp last punch date of the pay period

.IsLastPunchWeek last punch date of the week

.PunchSets Returns how many punch sets are in the day

.DeptHours Returns the hours allocated to a particular labor prompt with a particular value Example: ReportingDate.DeptHours("LaborPromptFieldName","LaborPromptValue")

.TotalHours([Category],[Date]) hours in the day (or date)

Examples: ReportingDate.TotalHours("Regular")

ReportingDate.TotalHours("", ReportingDate.Date + 1)

TotalHours function cannot get the total hours of a day that is before the

pay period of the punch that the script is running against

.TotalHoursOt hours in the day that are overtime eligible

.HoursToDate([Category]) hours to date in the week

.HoursToDateOt hours to date in the week that are overtime eligible

.WeekHours([Category]) hours in the week

Example: ReportingDate.WeekHours("Sick")

.PpHours([Category]) hours in the pay period

Example: ReportingEate.TotalPp("Tips")

.TotalWeek([Category]) Returns amounts for both numeric prompts and dollar amounts in the work week,

can take a category name or a field name of a numeric prompt

.TotalPp([Category]) Returns the amounts in both numeric prompts and dollar amounts in the pay period, can take a category name or a field name of a numeric prompt

.TotalDay([Category]) Returns the amounts in both numeric prompts & dollar amounts on the day of the punch, can take a category name or a field name of a numeric prompt

.SchedLines Returns the number scheduled lines on that day

.SchedHours([SchedLineNum]) Returns the amount of scheduled hours on the date, can take an integer number representing the scheduled line number.

.SchedIn([SchedLineNum]) Returns the in time of the first scheduled line on the date, can take an integer number representing the scheduled line number.

.SchedOut([SchedLineNum]) Returns the out time of the first scheduled line on the date, can take an integer number representing the scheduled line number.

.SchedPlace([SchedLineNum]) Returns the place of the first scheduled line on the date, can take an integer number representing the scheduled line number.

Script Box: **Accrual**

AccrueUp( BucketName, Amount, MaximumCap, VestingDate, ExpirationDate )

BucketName Required

Amount Required

MaximumCap Optional

VestingDate Optional

ExpirationDate Optional

Accrues Up (adds) the specified Amount of hours into the chosen bucket

Example: AccrueUp("PTO", 4, 160)

AccrueDown( BucketName, Amount, MinimumForBucket )

BucketName Required

Amount Required

MinimumForBucket Optional

Accrues Down (subtracts) the specified Amount of hours from the chosen bucket

Example: AccrueDown("PTO", 5)

GetBalance( BucketName ) Returns the Balance of the hours in the bucket

Example: GetBalance("Hours")

SetBalance( BucketName, Balance ) Sets the balance of the bucket

Example: SetBalance("Hours",0)

Script Box: **Overtime**

NonOtDay Non-Overtime Hours for day before Overtime is calculated

Example: NonOtDay=8

NonOtWeek Non-Overtime Hours for week before Overtime is calculated

Example: NonOtWeek=40

Ot1Threshold Returns weekly overtime threshold

Ot2Threshold Returns weekly doubletime threshold

Ot1DayThreshold Returns daily overtime threshold

Ot2DayThreshold Returns daily doubletime threshold

OTDayNumber Returns consecutive days worked

Some things to remember when using OTDayNumber are:  
1. The system looks at the work week only, not the pay period - what this means is that the days will only number 1-7, when the work week ends, the counter starts over.   
2. The system must be in the phase of processing the overtime calculations. In order to do this we need write this script inside a DURING statement.  
  
The system can evaluate against the OTDayNumber using greater-than, less-than, and equal to. Example: if(OTDaynumber > 5) would evaluate to TRUE for any hours worked on the 6th and 7th consecutive and days; if(OtDayNumber = 7) would evaluate to TRUE only on the 7th consecutive day worked.

OtRules(OtRule) Processes one of the rules found in the overtime category of processing rules

OtRule is one of the following:

AB - Sets weekly threshold for overtime at 42 hours for OT1 (Alberta Overtime)

C2 - Sets daily thresholds for overtime at 8 hours for OT1 and 12 hours for OT2 (California Overtime - 7 Consecutive)

CA - Sets daily thresholds for overtime at 8 hours for OT1 and 12 hours for OT2 (California Overtime)

CC - Overtime after 10 hours in a day or 60 in a week

CG - Sets daily thresholds for overtime at 10 hours for OT1 and 12 hours for OT2 (California Agricultural Law standards)

CM - Overtime after 11 hours in a day or 40 in a week (California make up law)

CO - Daily overtime at 12 hours

CR - Sets weekly thresholds for overtime at 40 hours for OT1 and 44 hours for OT2 (California Residential Child Center)

DH - Sets daily thresholds for overtime at 8 hours for OT1 and 12 hours for OT2 (California Overtime - 6/7 Consecutive)

DT - Sets daily thresholds for overtime at 8 hours for OT1 and 10 hours for OT2

LD - Overtime after 7 consecutive days of work, regardless of hours worked

LU - Weekly overtime on 7th day of workweek, and 6th day if Union

N1 - Sets daily thresholds for overtime at 8 hours for OT1 and 12 hours for OT2

N2 - Sets daily thresholds for overtime at 8 hours for OT1 and Weekly hours for 40 (Nevada Overtime)

NO - Sets it to straight time only, no OT or DT.

NV - Sets daily thresholds for overtime at 8 hours for OT1 (Nevada minimum wage override). Note that this only considers 8 hours when posted to a single calendar day, consider NV11 if rule applies to any 24-hour period.

ON - Sets weekly threshold for overtime at 44 hours for OT1 (Ontario Overtime)

ot4test - Test rule: OT after 4 hours in a day

OTThreshold - Imitates the overtime rules from SwipeClock, which were selected by a two-letter code. +

OTThresholdScript - This rule allows manipulation of overtime thresholds through scripting. +

P8 - Overtime after 8 hours in a day or 80 in a pay period

PI - Weekly overtime at 40 hours and double time at 56 hours

PP - Custom configuration of daily and weekly overtime thresholds (zero values denote immediate promotion to overtime)

S6 - Sunday double time after 6 hours

X2 - Sets daily thresholds for overtime after 12 hours in a day, no double time

XH - Sets daily thresholds for overtime at 10 hours (Double time not allowed)

+ Note the below rules requires scripting:

OTThreshold - Imitates the overtime rules from SwipeClock, which were selected by a two-letter code.

OTThresholdScript - This rule allows manipulation of overtime thresholds through scripting.

OtRules - This rule allows one to use the above two letter code above in a script.

The following are Overtime rules that cannot be accessed with the OTRules function:

HH - Sets daily thresholds for overtime at 8 hours (Double time not allowed)

NV11 - Applies overtime to any occurrence of more than 8 hours worked in any 24 hour period. Also applies OT after 40 hours in a workweek.

OT40 - STANDARD OVERTIME RULE, OVERTIME AFTER 40 HOURS PER WEEK

OT80 - Enables overtime calculation based on a pay period instead of a work week.