FrameReady Formulas

Each of these formulas generates a price per foot for moulding

Formulas	What it does	How to read it
I call this the "Arbitrary" option. Use it when you want to price the moulding differently than your regular mark-ups. It's an arbitrary thing; you look at it and say, "I know I can sell this for more than $16.00" So just enter the price you want to charge per foot for this one moulding.		
22.95	Arbitrary retail price per foot. Perfect for mouldings which you know can be sold for more than your usual markup.	Twenty-two dollars and ninety-five cents per foot.
This is a Straight Markup. It is usually used with chop or join pricing. The theory behind it, is that you know what your best markup is and you charge the same markup on all moulding regardless of the wholesale cost.		
cost * 5 field name operator number	An equation to calculate the retail price per foot value (number or amount)	"cost" is the name of the length field. "*" is the multiplication sign like "x". The "number" indicates how much the dollar amount in the cost field will by multiplied by.
The advantage of the "Round" formula is that it will ensure that the price calculates to two decimal points. This is useful if your multiplier has more than two decimal places.		
Round (cost * 4.835, 2) function equation # of digits	Rounds amount up to the decimal number indicated in the formula. The equation must be enclosed in parentheses (round brackets).	"Round" tells the type of function to be performed. The number "2" tells how many digits to which the number will be rounded up.
The benefit of this formula is that it evaluates the field to ensure that it actually *has* a cost in it. Remember that anything multiplied by zero (or a blank field) is still zero. This formula should not be used alone; it should be used with the "case" formula (shown below) which includes other options *in case* the cost field is blank.		
cost > 0; cost * 5	Looks to see if there is an amount in the cost field before calculating the equation (of cost times 5).	If the **amount in the cost field** is **greater than zero**, **then** (;), take the **cost** and **multiply** it by **5**.
The "Case" formula is used whenever you have more than one option. If the supplier has not provided us with all three wholesale costs, this formula will ensure that FrameReady will still be able to calculate a retail price per foot. It looks to a second, and third choice, *in case* the wholesale cost field is blank in your first option.		
Case (function chop > 0; chop * 3, cost > 0; cost * 4, join > 0; join * 2) equation	"Case" is used to show multiple options. The equation must be enclosed in parentheses (round brackets). The order indicates which field to look in first. This example shows all 3 pricing options: 1) chop 2) cost 3) join	**If** (case) the amount in the **chop** field is **greater than zero**, **then** (;), take the amount in the **chop** field and **multiply** it **by 3**. **Otherwise** (,) look in the **cost** field. And, if the amount is **greater than zero**, **then** (;), take the amount in the **cost** field and **multiply** it **by 3**. **Otherwise** (,) look in the **join** field...

The "Sliding Scale" formula operates on the principle that lower priced moulding can have a higher markup than more expensive moulding. Most commonly used with Length pricing and sometimes chop. Watch that your markups are close in sequence. Eg. 3, 2.95, 2.90, 2.85, etc. Gaps in markups; cost you money.

Case (function cost < 4.00; cost * 5.00, cost < 6.50; cost * 4.95, cost < 9.00; cost * 4.90, cost < 15; cost * 4.8, cost < 20; cost * 4.7, cost < 25; cost * 4.6, cost * 4.5) equation	This equation produces a sliding scale markup for one 'Cost Type' -- in this case it is cost (Length). The 'Case' function provides the multiple option of different mark-ups for each price range listed. The last line is basically a 'catch all' for everything else.	**If** (case) the amount in the **cost** field is **less than four** dollars, **then** (;), take the amount in the **cost** field and **multiply** it **by five**. **Otherwise** (,) if the **cost** is (above 4 but) **less than 6.50, then** (;), take the amount in the **cost** field and **multiply** it **by 4.95**. The last line reads: and everything else will have the cost multiplied by 4.5.

A slight variation on the "Sliding Scale" formula (above) setting a minimum retail price per foot.

Case (minimum per foot cost < 4.00; 20, cost < 6.50; cost * 4.95,	This equation produces a sliding scale but covers a mimimum charge for less expensive mouldings.	**If** (case) the amount in the **cost** field is **less than $4**, **then** (;), the **retail $ per foot** is $20.00. Otherwise, if...

The "Cost Type" formula is used when you want to charge your customer a price based on how you order the moulding: Length, Chop or Join. Best for companies with all 3 cost types; blanks = zero The price is calculated based on which radio circle is marked. This can be changed directly on the order screen for rush jobs.

Case (field names cost >0 and Cost Type = "Length"; Cost * 5, chop > 0 and Cost Type = "Chop"; Chop * 3; Join > 0 and Cost Type = "Join"; Join * 2) operators	This equation should identify all 3 Cost Types: Length, Chop or Join. Pricing is determined by which radio circle is marked for the item. (Usually used to identify how the moulding is ordered.) Does not matter in which order the cost types are listed. Can also be used in combination with sliding scale pricing and other formulas.	**If** (case) the amount in the **cost** field is **greater than zero** dollars and the circle for **Cost Type** is marked as **Length**, **then** (;), take the amount in the **cost** field and **multiply** it **by five**. **Otherwise** (,) if the amount in the **chop** field is **greater than zero** dollars and the circle for **Cost Type** is marked as **Chop**, **then** (;), take the amount in...

The "Ceiling" formula can be beneficial when suppliers round up the footage or send extra footage. (You may also want to use the Minimum Footage field and the Oversize and Surcharge fields.)

Ceiling (function Case(Join>0; Join * 2, Chop > 0; Chop * 3.5, Cost > 0; Cost * 5))	This equation creates a ceiling by rounding the calculated dollar per foot up to the nearest whole number. E.g.. $24.13 will become $25.00 per foot.	**Round up to an even dollar amount** (Ceiling) and **if** (Case) the amount in the **join** field is **greater than zero**, **then** (;), take the amount in the **join** field and **multiply** it **by 2**. **Otherwise** (,) look in the **chop** field. And, if the amount is **greater than zero**...

This is a combination of the “Sliding Scale” formula and the “Straight Markup” formula. This allows you to cover all 3 cost types and define multiple options for each. Great for combining several pricing options into one formula.

```
Case (
Chop>0;
  Case (
   chop<12; chop * 2.5;
   chop<18; chop * 2.45;
   chop<24; chop * 2.40;
   chop *2
   );
Join>0; Join * 2,
Cost > 0; Cost * 4
)
```

function

This allows a sliding scale with all cost possibilities covered.

This equation has two “Case” functions. This means that within one option, there are other options. E.g.. 1st option: choose Chop, or Join, or Cost. 2nd option: If you choose Chop, choose under 12, under 18, under 24 or everything else.

If (case) the amount in the **chop** field is **greater than zero**, **then** (;), do the following. **If** (case) the amount in the **chop** field is **less than 12**, **then** (;) take the amount in the **chop** field and **multiply** it **by 2.5**. **Otherwise** (;) if the amount in the **chop** field is **less than 18**, **then** (;) take the amount in the **chop** field and **multiply** it **by 2.45**. **Otherwise** (;) if the amount...

A minor variation on previous formulas. A dollar amount is added to the retail per foot price in addition to the markup or multiplier.

```
Case(
  Chop>0;
    Case(
    chop<12; chop * 2 + .90;
    chop<20; chop * 1.99 + .90;
    chop *2 +.90
    );
  Join>0; Join * 2 +.90,
  Cost>0; Cost * 5 + .90
  )
```

operators

This equation adds a dollar amount onto the per foot price. (This is different from the Set Price field in FrameReady where a dollar amount is added to the price of the Frame.)

If (case) the amount in the **chop** field is **greater than zero**, **then** (;), do the following. **If** (case) the amount in the **chop** field is **less than 12**, **then** (;) take the amount in the **chop** field and **multiply** it **by 2** and **add** (+) **90 cents**. **Otherwise** (;) if the amount in the **chop** field is **less than 20**, **then** (;), take...

Tips:

Spaces and carriage returns do not affect your formula, they simply make it easier for your eyes to read it.
Fields and Formulas must be spelled correctly.
For every opening bracket or parentheses (you must have a closing bracket or parentheses) .
At the end of an equation, you may use either a comma or a semi-colon. However, only the semi-colon may be used between the definition of “when” it is used and “what” is done.
It is recommended that all cost types (cost, chop & join) be covered in your moulding formulas.

SoftTouch Solutions Inc
4179 Petrolia Line, Ste. 1 Petrolia ON N0N 1R0
888-281-3303
www.frameready.com
softouch@mnsi.net

The “Let” formula is the most advanced and comprehensive. Pricing is based on the radio circles. It allows you to arrange your sequence of pricing options based on which circle is selected. Eg. If Length circle is marked, the sequence is: Length, Chop, Join. If Join circle is marked, sequence is: Join, Chop, Cost.

```
Let([                          name of equation
Length Price =
    Case(
        Cost<.99; 15;
        Cost<1.49; Cost * 5;
        Cost<1.99; Cost * 4.95;
        Cost <2.99; Cost * 4.90;
        Cost * 4.85);
Chop Price = Chop * 3;
Join Price = Join * 2
];

Case(
Cost Type = “Length”;
    Case(
        Cost>0;  Length Price;
        Chop>0;  Chop Price;
        Join>0;   Join Price);
Cost Type = “Chop”;
    Case(
        Chop>0; Chop Price;
        Join>0; Join Price;
        Cost>0;  Length Price);
Cost Type = “Join”;
    Case(
        Join>0; Join Price;
        Chop>0; Chop Price;
        Cost>0; Length Price)
)
)
```

The Let [ ] brackets identify the pricing formulas to be used. The “Let” formula allows us to give a title or name to an equation, in this case “Length Price”. Below the square brackets, is the condition of when “Length Price” is to be applied. This area identifies the sequence for selection.

SEQUENCE:
If the radio circle is marked as length...
and there is an amount in the field...
then, my first choice is to price by what we have named “Length Price”, then Chop Price, and lastly Join Price.

However, if the radio circle for Cost Type is marked as chop...

PRICING FORMULA: If you use “Length price”, then do the following... **If** (case) the **cost** is **less than .99**, **then** (;) the retail price per foot is **$15**. Otherwise (;) the wholesale **cost** is **less than 1.49**, **then** (;) take the amount in the **cost** field and **multiply** it **5** times. Otherwise, ...

If (case) the circle for **Cost Type** is marked as **Length**, **then** (;) do this...
If (case) the **cost** field is **greater than zero**, then (;) **Price by “Length Price”** *(formula for “Length Price” shown above)*. Otherwise (;), **if** the **Chop** field is **greater than zero**, then (;) **price by “Chop Price”** *(formula for “Chop Price” shown above)*. Etc...

Used in Equation:	Identified As:	Read As:
Case	name of function	If
;	semicolon	then
,	comma	otherwise
cost	name of field for Length wholesale	Length
chop	name of field for Chop wholesale	Chop
join	name of field for Join wholesale	Join
Let	name of function	gives a formula or equation a title, name or expression.

Operators

An Operator is a symbol or instruction that manipulates expressions in a formula. For example, the plus (+) operator tells FileMaker Pro to add one expression to another. FileMaker Pro has four types of Operators.

Mathematical Operators

Symbol	Name	Definition
+	Plus	Adds two values
-	Minus	Subtracts the second value from the first
*	Multiply	Multiplies each value
/	Divide	Divides the first value by the second
( )	Precedence	FileMaker Pro evaluates formulas from left to right, performing multiplication and division before addition and subtraction. Using parentheses lets you change the order: FileMaker Pro evaluates expressions between parentheses first.

Comparison Operators

Symbol	Name	Definition
=	Equals	True if the items are equal
>	Greater Than	True if the value on the left exceeds that on the right
<	Less Than	True if the value on the left is less than the value on the right
>=	Greater Than or Equal To	True if the value on the left is greater than or equal to the value on the right
<=	Less Than or Equal To	True if the value on the left is less than or equal to the value on the right

Logical Operators

Symbol	Name	Definition
AND	and	True only if both items are true
OR	or	True if either item is true
XOR	'this or that'	True if either of the expressions (but not both) is true
NOT	'switch'	Changes the value from False to True or from True to False

Text Operators

Symbol	Name	Definition
" "	text constant	Marks the beginning and end of characters to be considered as text. Quotation marks without text between them indicates an empty value (no text). If you enter text into a formula without using quotation marks, FileMaker Pro tries to interpret the text as a 'Field' or 'Function' name.
\	backslash	Identifies that an operator character will be used as a character instead of an operator. Eg. "\"Fred\" and I" will read, "Fred" and I
&	Concatenate	Appends the text string on the right to the end of the text string on the left (joins the right and left together.) Text strings can be fields, constants enclosed in quotes, or certain functions.