Formación en programación de problemas WebWorK

Grupo LEMA, 2021

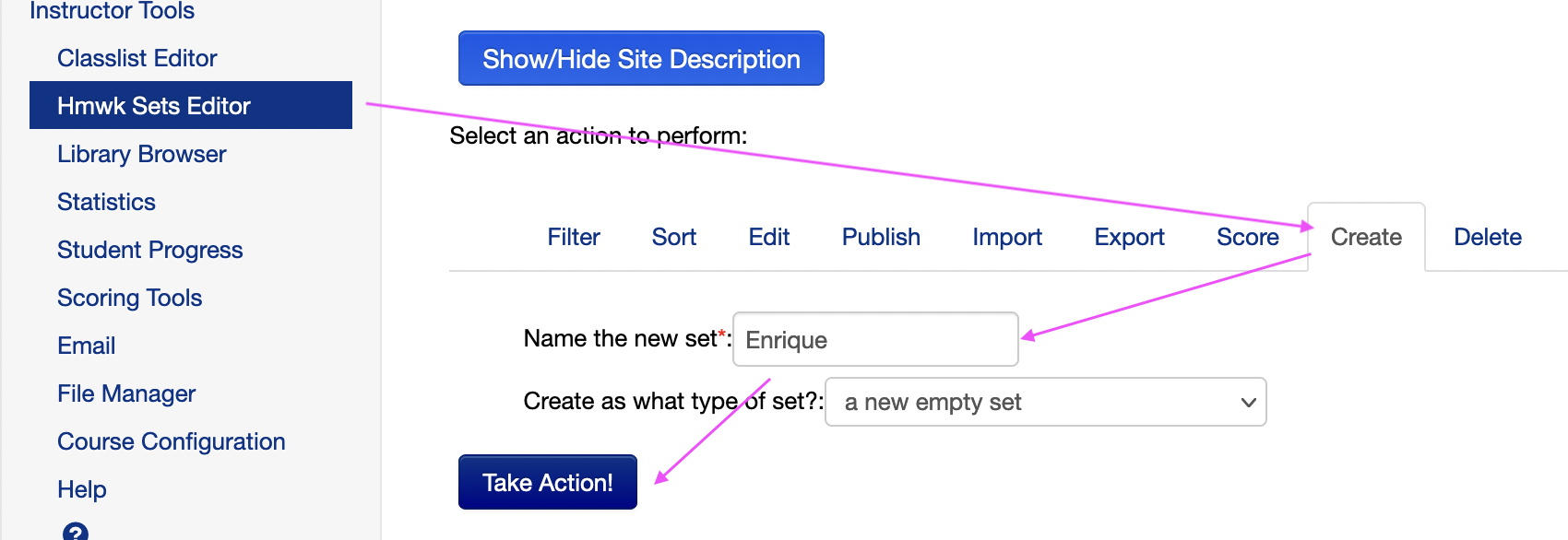
www.grupolema.org

Licencia de uso cc-by 4.0

Documento con código de archivos pg que se mencionan en este documento: [ArchivosPG](https://docs.google.com/document/d/1Ju-F9Tu87e6XYgIPOaN5qGyapih149d0KaSbkBZp6vs/edit#)

# 1. Crear un "blank problem" y entrar a editarlo

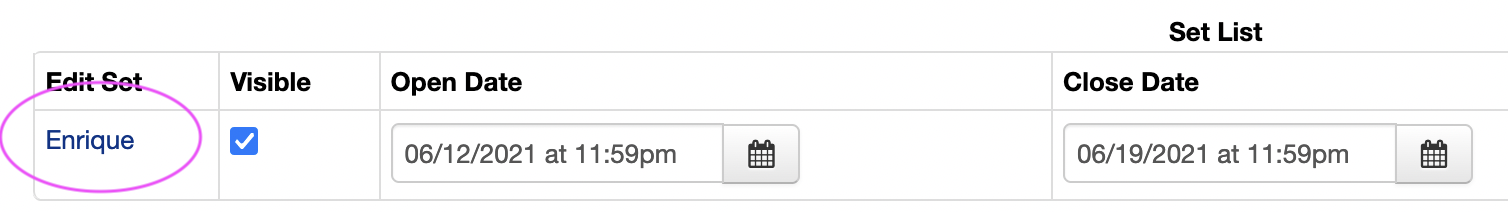
1. Crear una tarea con su nombre.



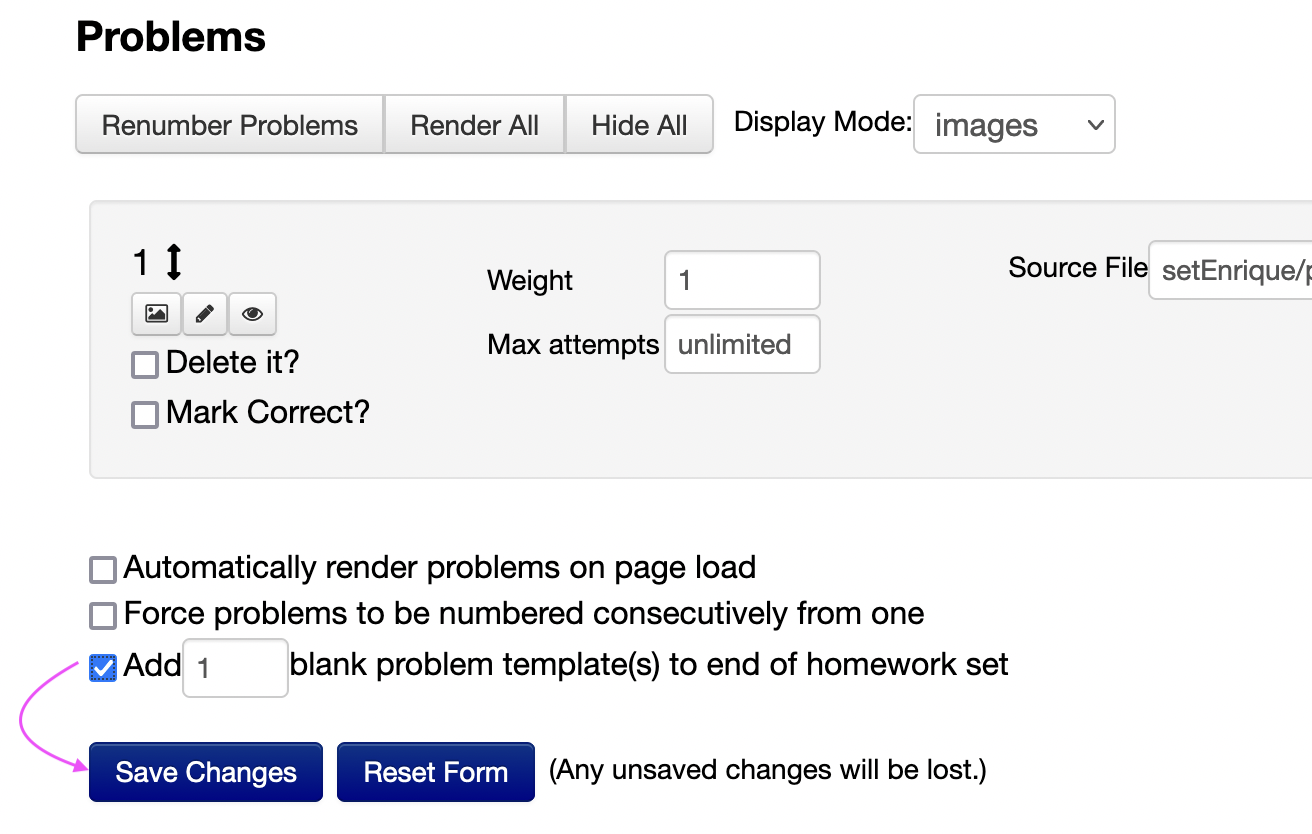
2. Entrar a editar la tarea ("edit set", que salga la lista de problemas)



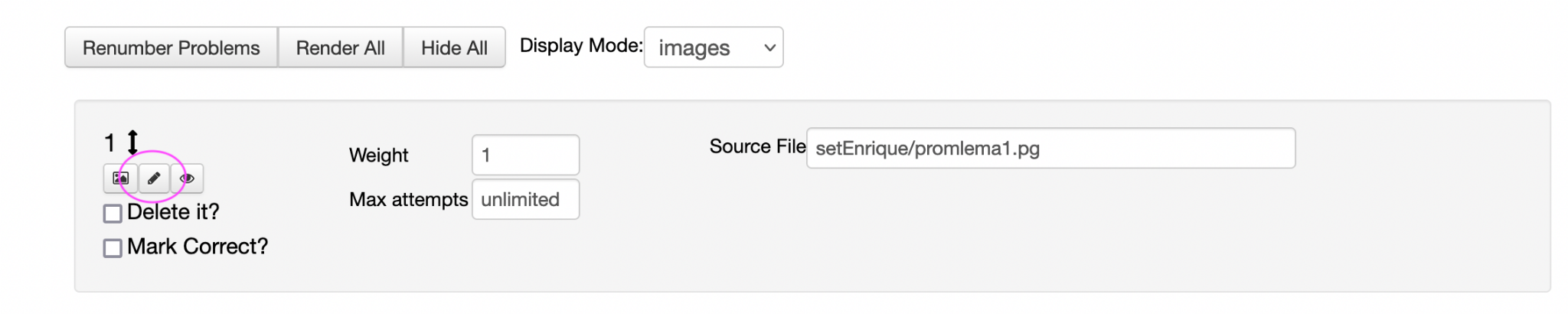
seguido de...



3. Agregar un blank problem a la tarea:



4. Entrar a editar el nuevo problema con el botón del lápiz:

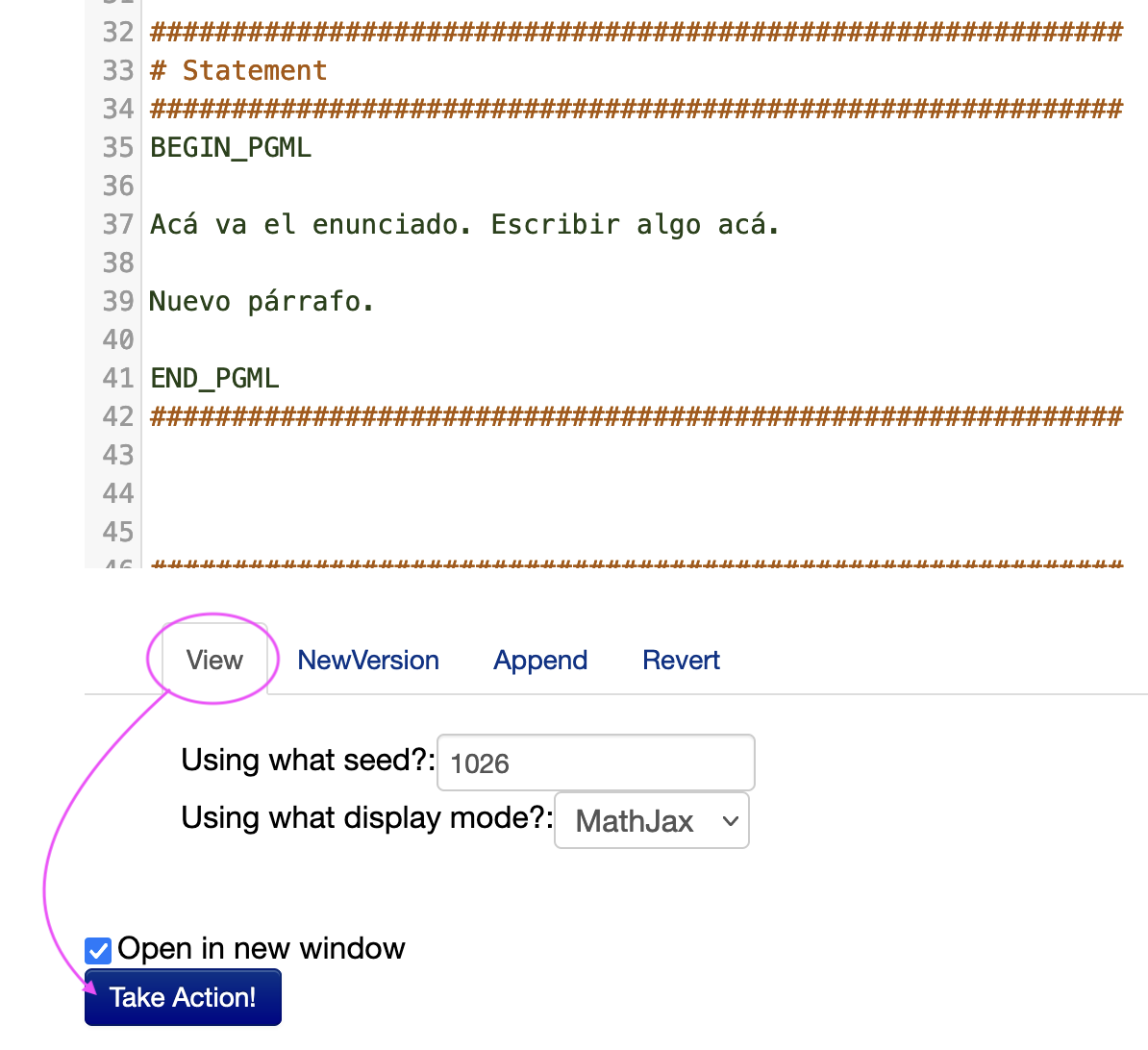


# 2. La estructura básica de un archivo .pg (usando PGML)

* Archivo barebonesEstructuraBasica.pg
* Hay espacios en los que el lenguaje es Perl, otros en los que es PGML... es un archivo "híbrido".
* Matemáticas con LaTeX:
  1. # LaTeX math-mode ($...$): [`\LaTeX`]
  2. # LaTeX displaystyle ($\displaystyle...$): [``\LaTeX``]
  3. # LaTeX en display-mode ($$...$$): [```\LaTeX```]

# 3. Vista previa de los cambios

* Preview con el botón "ver---> ejecutar" (View --> Take Action)
* Ejercicios básicos de escribir mate en statement, poner hint, escribir solución.

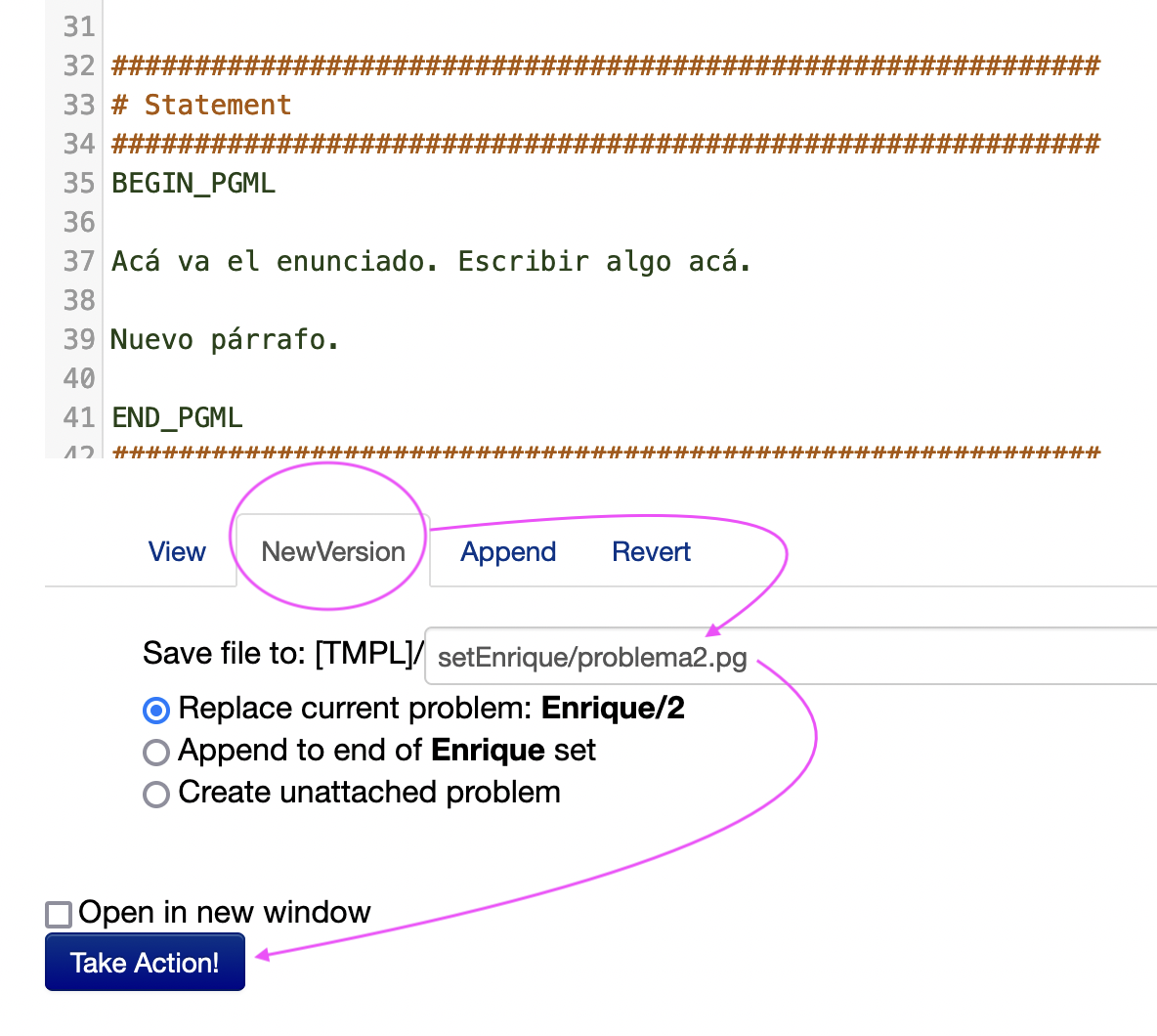


# 4. Crear un nombre apropiado para el archivo y guardar los cambios

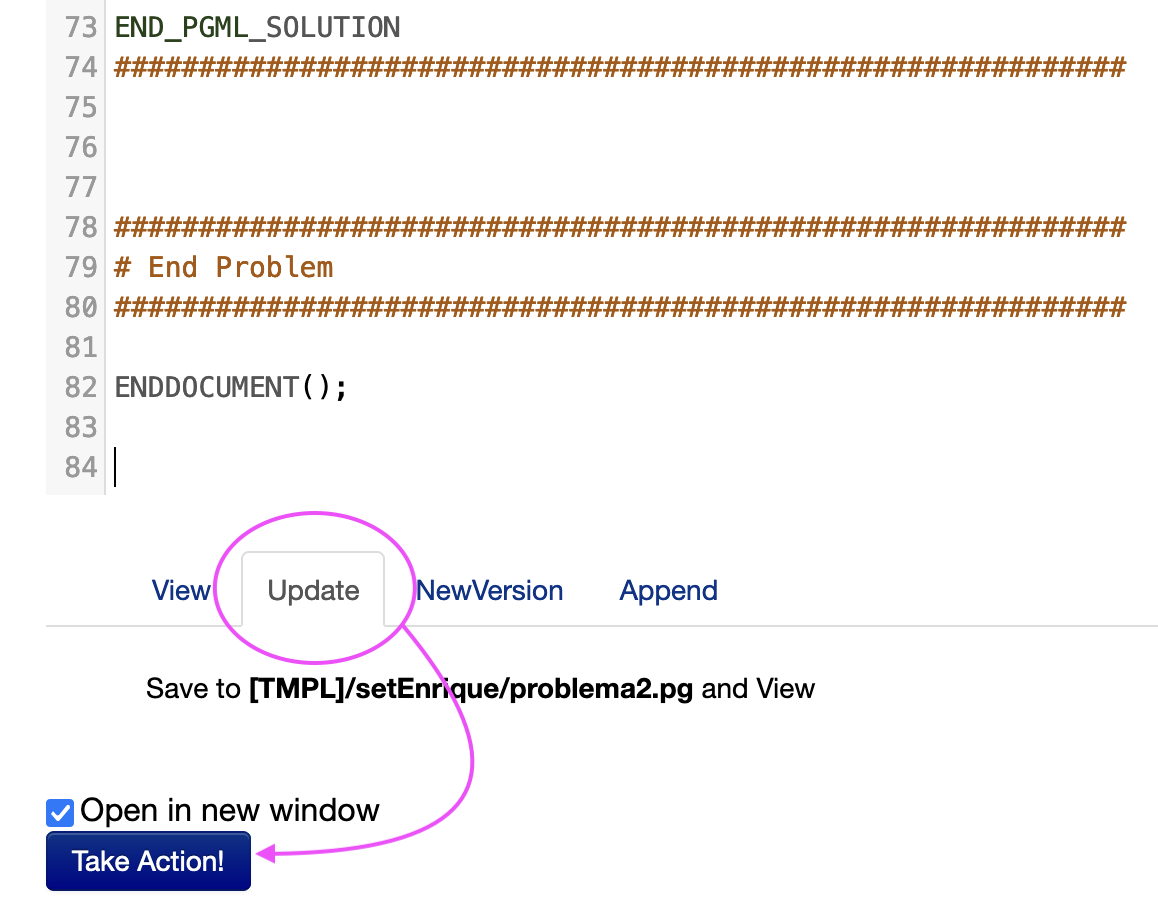
Hay que darle un nombre al archivo (que no quede como el default *blankProblem.pg* porque eso causa problemas).

Para hacer esto:

Nueva versión ---> un buen nombre (e.j. setEnrique/problema1.pg)

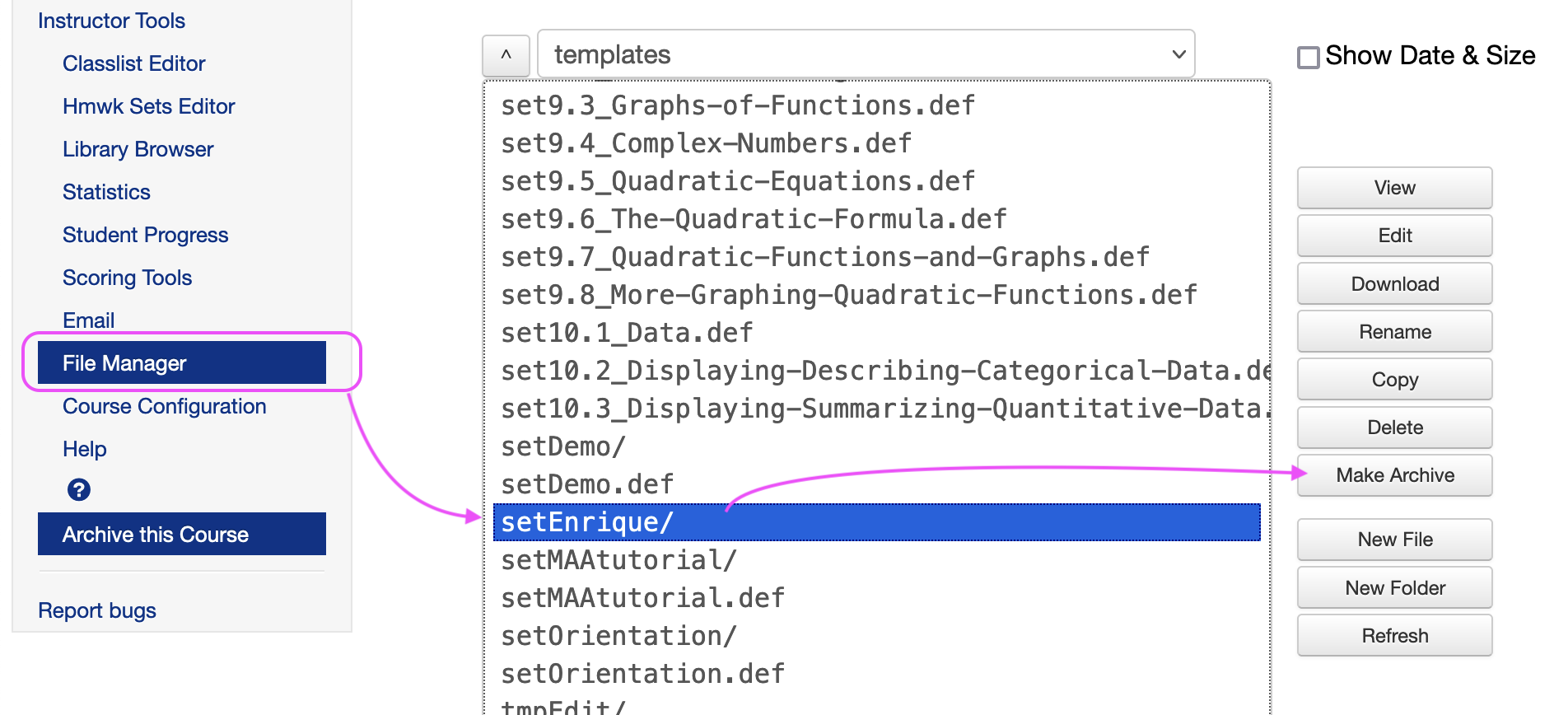


# 5. Como guardar cambios una vez el archivo tenga el nombre correcto

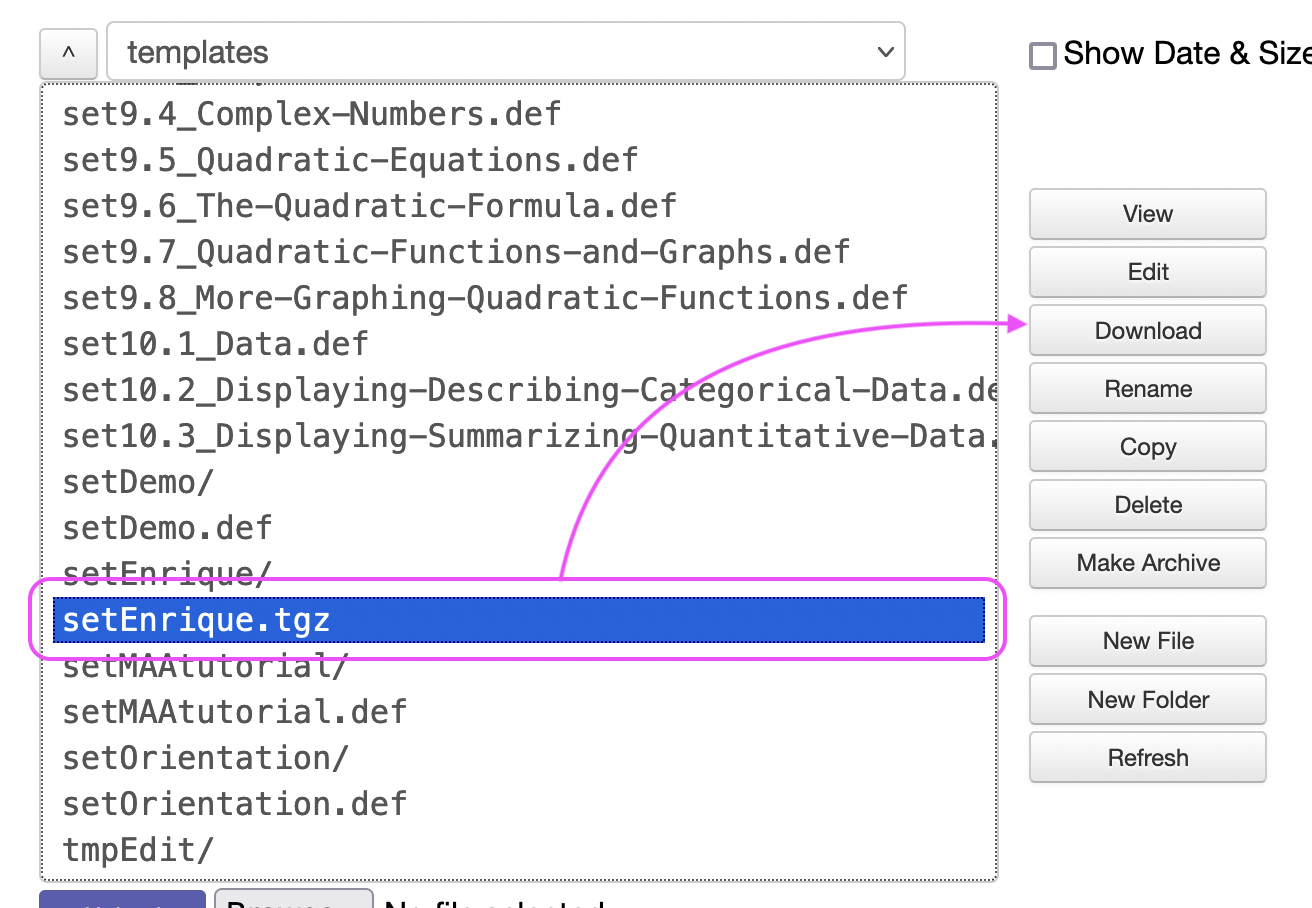


# 6. Descargar los archivos .pg

1. Ir a admin archivos y crear "archive" (tgzip)



2. Seleccionar el tgzip que se creó y descargarlo.



# 7. Campos de respuesta y cómo programarlos

**Los contextos, el "Compute", el objeto de respuesta.**

barebonesCampoRespuesta.pg

* Escondido en el hint hay ejemplos de cómo:
  + cambiar el tamaño de campo de respuesta y
  + agregar ayudas de ingreso de respuestas "AnswerFormatHelp.pl".
* Ver la respuesta correcta como la va a ver el estudiante chuleando "Correct Answers" antes de presionar "Check Answers"

**Números y tolerancias**

barebonesCampoRespuestaTolerancia.pg

**Expresiones algebraicas**

* constructos y evaluación: barebonesExpresionesConstructos.pg
* variables distintas a x (ajustes a contextos): barebonesExpresionesVariables.pg
  1. Más info de contextos: https://webwork.maa.org/wiki/Introduction\_to\_Contexts
* dominios para cambiar equivalencias algebraicas (y más sobre ajustes a contextos - deshabilitar funciones y operadores): barebonesExpresionesDominio.pg
* más info sobre expresiones:   
  https://webwork.maa.org/wiki/Formula\_(MathObject\_Class)

**Puntos**

barebonesPuntos.pg

**Listas (pueden ser de puntos, expresiones, números, etc.)**

barebonesListas.pg

**Más info sobre contextos y math objects**

https://webwork.maa.org/wiki/Introduction\_to\_Contexts

https://webwork.maa.org/wiki/Category:MathObject\_Classes

**Selección múltiple**

* Selección Múltiple: barebonesSelMultiple.pg
* Dropdown: barebonesDropdown.pg
* Ejemplo con pg (se ve bastante más complejo): https://webwork.maa.org/wiki/MultipleChoiceProblems

**Otros MathObjects:**

intervalos, complejos, vectores, matrices, conjuntos, cadenas,

https://webwork.maa.org/wiki/Category:MathObject\_Classes

# 9. Aleatorización

**La parte del algoritmo**

* barebonesRandomizacion.pg
* Para ver distintas versiones ---> cambiar seed y volver a presionar el botón para ver el problema "ver---> ejecutar" (View --> Take Action)
* Ejercicio: Programar: "Soluciona z^2=25" con aleatorización.

**Pitfalls**

Apenas uno se mete con valores aleatorios comienzan a aparecer problemas como 1x, 4+-3x, 0x, x^1,...

* barebonesRandomizacionPitfalls.pg
* barebonesRandomizacionPitfalls2.pg
* Más info sobre el "reduce" en: https://webwork.maa.org/wiki/Reduction\_rules\_for\_MathObject\_Formulas

**Uso de condicionales para evitar aleatorizaciones problemáticas**

barebonesRandomizacionCondiciones.pg

**El error básico cuando se está aleatorizando:**   
El error es empezar a programar un problema pensando en la aleatorización y no en "¿Cómo se debería ver el problema en WW?¿Cuáles y cuántos son los campos de respuesta que voy a crear? -- ¡programar un problema en WW no es escribir el algoritmo de aleatorización!

**Estar pendiente de cómo se ven las respuestas para los estudiantes**

Recomendación: revisar siempre como están viendo los estudiantes la respuesta. El constructo "Compute" en vez de especificar el tipo de objeto en la variable de respuesta a veces es útil.

* barebonesRandomizacionRespuestasParaEstudiantes.pg
* Link de referencia (explicando el "Compute"): https://webwork.maa.org/wiki/Introduction\_to\_MathObjects

# 10. Partes a,b,c

* barebonesPartesabc.pg
* En partes que no se pueden ver hasta que se haya respondido la parte anterior correctamente:   
  barebonesPartesSeparadas.pg

# 8. PGML vs PG

* Wiki: https://webwork.maa.org/wiki/Authors
* subject area templates, ODE  
  https://webwork.maa.org/wiki/GeneralSolutionODE1
* subject area templates, Calculus, linear approximation  
  https://webwork.maa.org/wiki/LinearApprox1
* OPL y cómo uno puede "clonar" cualquier problema.

# 11. Imágenes [falta]

* Estáticas - prácticas de fólderes asociados.
* Estáticas pero aleatorizadas
* Aleatorizadas - sistema básico
* Aleatorizadas - tikz en WW 2.16

# 12. Recomendaciones generales al programar

* Tratar de hacer el pensamiento de estudiante visible con campos de respuesta (no solo la respuesta correcta).
* Hay que pensar en todas las respuestas aceptables, y aceptarlas todas como correctas. Ejemplos:
  + Tolerancia numérica para tener en cuenta redondeos de números en cálculos intermedios, o advertirle al estudiante que solo se va a aceptar la respuesta exacta.
  + Uso de constantes +c (en integración). ¿Otras letras para las constantes?
* Hay que ser claro con el estudiante en qué se espera.
  + "Usa C como constante de integración."
  + "Escribe la respuesta exacta, o aproxima a la milésima más cercana." (puede que en este caso el campo de respuesta califique a menos decimales, por redondeos intermedios).
  + "Usa $r$ para el radio en tu respuesta."
* La aleatorización se piensa después de pensar los campos de respuesta, no antes.
* La aleatorización no puede tener versiones más complejas para los estudiantes.
* Los Hints son importantes (a menudo sirve dar una versión del principio de la solución escrita como hint). Las soluciones completas sirven si el profesor las va a hacer disponibles.

# 13. Revisión de problemas programados por otros para que queden listos para uso

* Revisar cómo se ven la respuestas (check "CorrectAnswers" antes de presionar el botón de "CheckAnswers")
* Revisar que no haya errores causados por la aleatorización en los textos que ve el estudiante (pitfalls): revisar varios seeds.
* Revisar que el problema acepte todas las posibles respuestas y formas de respuesta válidas. Revisar varios seeds.
* Revisar que el problema no marca respuestas incorrectas como correctas.

# 14. Ejemplos más complejos

* mostrar números sin decimales:
  + Context()->flags->set(reduceConstants => 0, reduceConstantFunctions => 0);
    - $f = Formula("(1+sqrt(5))/2)");
    - in which case $f would remain (1 + sqrt(5))/2 rather than the usual 1.61803.
* evaluar dos campos al tiempo, o en distinto orden
* StressTests (Consortium)
  + 1. Question with algorithmic graph in the statement
  + 2. Question with algorithmic graph in the statement:
  + 3. A question in which the student answer can have an arbitrary letter as a constant
  + 4. A question that grades a symbolic expression involving a function and a hint
  + 5. One that has an angle answer field which grades mod 2pi
  + 6. Questions with tolerance for coefficients in expressions
  + 7. Grade depending on other student responses
  + 8. Check for a composition of functions
* Evaluar parametrizaciones de superficies (email Eric)

# 15. Un buen resumen corto:

* <https://github.com/drdrew42/WeBWorK-problem-authoring/blob/main/intro.md>