

<https://www.miyoushe.com/ys/article/46972789>

被锁了几次，我服了

Part 1: 介绍

原神伤害计算器的全部代码是由我所写，该项目已开源。允许所有人修改其中的代码并且不需要告知我，如需发布请注明作者并也应当开源(如有特殊需求可以私信我)。原神社区需要我们共同的维护。

为什么要做这个呢？

因为现在的原神工具箱大都很少支持自定义，伤害计算不够准确，也有很多的功能并没有实现。我希望的是一个高度自定义化又不失简洁性的准确的伤害计算工具。因此有了这个想法。

现在的进展如何？

本项目从上周开始准备并编写，于12月22日发布内测版 V1.0，于12月26日发布初步公测版 V1.1，欢迎大家使用并查错，提出修改意见。

目前实现的功能是 伤害计算，圣遗物自定义评分与预测。

未来要做的功能是 组队伤害便捷计算，圣遗物管理。

之后什么时候更新呢？

做完 V1.1 后确实没有太多时间了，先要备考期末，然后过年的时候既要走亲戚，还要完成大创结题。因此之后两个月更新会比较少，但等我有空的时候会继续更新的哦~

下载方法：

开源地址后缀是/virtualxiaoman/Genshin-Calculator，懂的应该都懂，不懂的等我看评论能不能发。

Part 2: 使用方法

点击 exe 即可启动。

界面 1: 伤害计算

界面如下：

原神伤害计算器

伤害

组队

圣遗物

AE

攻击:

2000

DB

增伤:

46.6

%

CD

暴击率:

80

%

ER

反应过程

☒ 水火蒸发

☐ 火水蒸发

☐ 火冰融化

☐ 冰火融化

DF

人物等级:

90

RT

抗性:

10

%

信率:

300

%

暴击:

180

%

基础伤害加成:

0

%

基础伤害加成:

0

%

计算数据

存储数据

读取数据

是否激化

☐ 激化

☒ 不激化

激化类型

☐ 原激化

☐ 超激化

☒ 蔓激化

激化反应加成值:

0

%

点击上方的按钮能够实现:

1. 计算数据: 根据界面上的数据计算

2. 存储数据: 存储当前界面上的数据到本地

3. 读取数据: 读取本地数据并更改当前界面内的数值

!!! 存储数据前请务必先计算数据一次, 不然会闪退!!!

在各个框里输入角色的属性，点击“计算数据即可”。如下图：

原神伤害计算器

伤害

组队

圣遗物

AE

攻击:

2001.1

DB

增伤:

172.4

%

CD

暴击率:

87.1

%

ER

反应过程

☐ 水火蒸发

☒ 火水蒸发

☐ 火冰融化

☐ 冰火融化

DF

人物等级:

90

RT

抗性:

10.0

%

信率:

283.25

%

暴击:

170.48000000000001

%

基础伤害加成:

0

%

基础伤害加成:

0

%

计算数据

存储数据

读取数据

是否激化

☐ 激化

☒ 不激化

激化类型

☒ 原激化

☐ 超激化

☐ 蔓激化

激化反应加成值:

0

%

期望伤害:

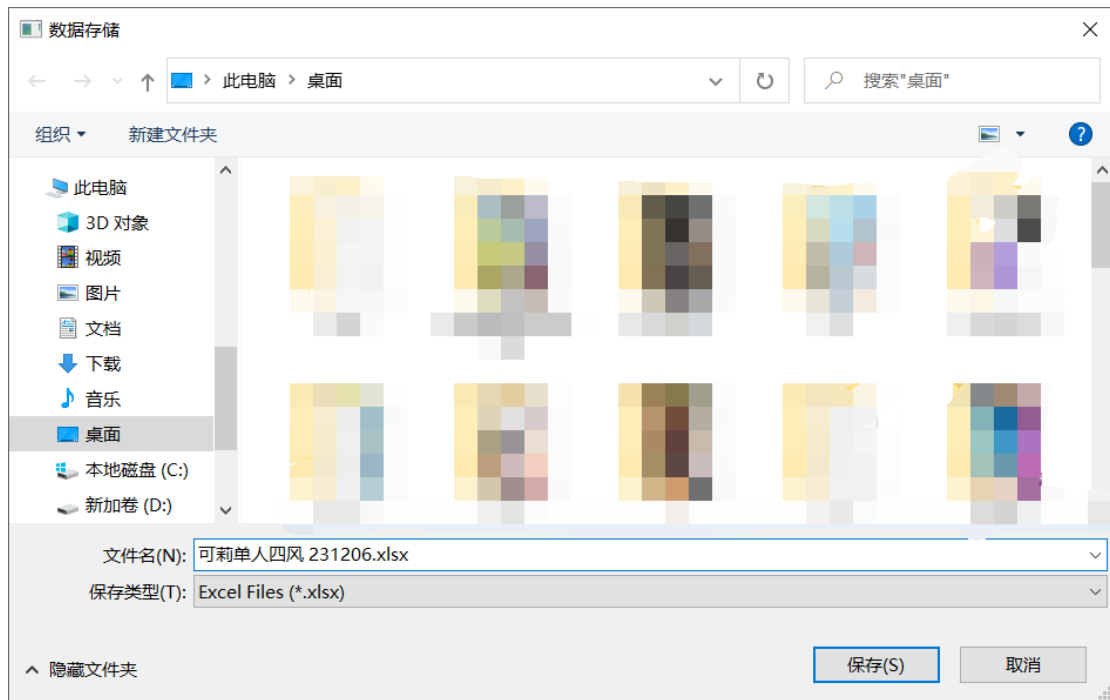
31788.870909578698

暴击伤害:

34602.27872348182

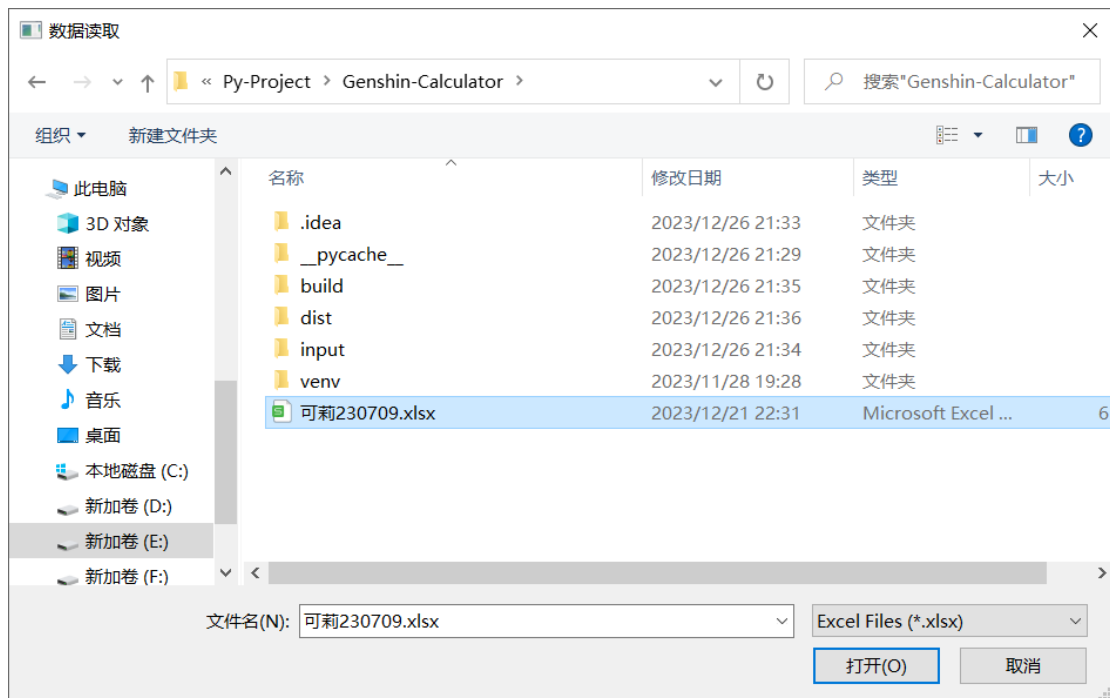
这里采用的数据是我的暑假的时候保存的一份可莉伤害计算。

如果需要存储这一份数据，请点击“存储数据”按钮：



建议选择你熟悉的文件夹，保存的文件名也应该比较好认，便于你的管理。在此页面点击“保存”即可。

读取数据也差不多：



选择之前保存过的数据，点击“打开”就能读入本地的 Excel 数据了。

界面 2： 还没做，开摆

界面 3： 圣遗物自定义评分与预测

这是初始界面：

伤害

组队

圣遗物

圣遗物评分与预测

原来的圣遗物

☐暴击率

☐大生命

☐大攻击

☐大防御

☐充能

☐暴击

☐大生命

☐大攻击

☐大防御

☐精通

☐暴伤

☐小生命

☐小攻击

☐小防御

☐精通

当前等级

☐+0

☐+4

☐+8

☐+12

☐+16

☒+20

点击“计算数据”后，这里会显示你的圣遗物词条数

点击“计算数据”后，这里会显示对圣遗物的预测

现在的圣遗物

☐暴击率

☐大生命

☐大攻击

☐大防御

☐充能

☐暴击

☐大生命

☐大攻击

☐大防御

☐精通

☐暴伤

☐小生命

☐小攻击

☐小防御

☐精通

当前等级

☒+0

☐+4

☐+8

☐+12

☐+16

☐+20

点击“计算数据”后，这里会显示你的圣遗物词条数

点击“计算数据”后，这里会显示对圣遗物的预测

对圣遗物的自定义加权分数

暴击：100

暴伤：100

大生命：80

小生命：50

大攻击：80

小攻击：50

大防御：80

小防御：50

充能：75

精通：75

计算数据

赌赢OR赌输

这里会提供一些建议，关于你是否应该赌圣遗物。仅供参考，不代表真实事件

如果两个圣遗物加起来大于或等于12级点右边的按钮，否则请点左边的

对于某个圣遗物，请选择它有的词条，并填写入输入框里，并给前面的的方框打钩：

点击“计算数据”是用来计算词条数的，

点击“赌赢 OR 赌输”是用来计算新胚子能赢过旧胚子的概率。

如下图：

主页面

功能2

功能3

圣遗物评分与预测

原来的圣遗物

☒暴击率

☒大生命

☐大攻击

☐大防御

☐充能

☒暴伤

☒小生命

☐小攻击

☐小防御

☐精通

3.9

5.8

7.8

1046

当前等级

☐+0

☐+4

☐+8

☐+12

☐+16

☒+20

词条数为：7.6544
暴击词条数:1.182
大生命词条数:1.172
暴击伤害词条数:1.182
小生命词条数:4.119
加权后的词条数为：5.3606
暴击词条数:1.182

已经是最高等级了

现在的圣遗物

☒暴击率

☒大生命

☐大攻击

☐大防御

☐充能

☒暴伤

☒小生命

☐小攻击

☐小防御

☐精通

2.7

5.8

5.4

299

当前等级

☒+0

☐+4

☐+8

☐+12

☐+16

☐+20

词条数为：3.9855
暴击词条数:0.818
大生命词条数:1.172
暴击伤害词条数:0.818
小生命词条数:1.177
加权后的词条数为：3.1625
暴击词条数:0.818

全部命中的概率是3.12%
预估强化后的加权词条数为：7.2875

对圣遗物的自定义加权分数

暴击：100

暴伤：100

大生命：80

小生命：50

大攻击：80

小攻击：50

大防御：80

小防御：50

充能：75

精通：75

计算数据

赌赢OR赌输

赌赢的概率是:0.9999799728393555

因为这个截图是以前截的，所以界面稍微有点不一样，但是这不影响

Part 3: 注意事项 以免出现闪退

在伤害计算时，请务必保证每一个框都已经填写了，即使你不知道填什么，就按默认值来就行。

在圣遗物自定义评分时，为什么要求两个圣遗物的等级之和要大于+12级？因为如果等级太小，强化后的结果过多，电脑会爆炸。比如一个初始四词条的圣遗物，经过强化后理论上有 $16 \times 16 \times 16 \times 16 \times 16 = 1048576$ 个结果（每次强化有四个词条，每个词条有四个成长值，所以是 16。五次方是因为强化五次）。两个拥有 1048576 个结果的圣遗物相比就有 10^{12} 次方次比较。其他的应该没啥能造成闪退的了。

Part 4: 代码逻辑

UI 部分不做介绍，这里仅说说计算部分。

伤害计算部分

实现伤害乘区学的代码：

```
def damage_cal(AE: float, DB: float, CD: float, ER: float, DF: float, RT: float) -> float:
    """
    根据六大乘区的伤害计算，这里暂时仅作为测试。
    :param AE: 基础伤害区 + 激化加成值 (攻击区 Attack × 倍率区 Damage Multiplier + 激化区 Elemental Reaction)
    :param DB: 增伤区 Damage Bonus
    :param CD: 暴击区 Critical Damage
    :param ER: 增幅区 Elemental Reaction
    :param DF: 防御区 Defense
    :param RT: 抗性区 Resistance
    :return: damage : 计算得到的伤害值
    """
    damage = AE * DB * CD * ER * DF * RT
    return damage
```

实现传入所有参数来计算伤害的代码：

```
import damage_calculator as dc

def damagecal_detailedly(atk=1304.1, talent=2.8325, em=500.6, catalyze_verify=1, damage_catalyze_increased: float = 0,
                        catalyze_type=2, added_basedamage: float = 0, multiply_basedamage: float = 0,
                        db_increased=0,
                        cr=0.458, cd=2.294, cr100=0,
                        elemental_magnification=1.5, increased_reaction_coefficient=0,
                        person_level=90, hilichurl_level=90, reduce_defenses=0, ignore_defenses=0, increase_defenses=0,
                        reduce_resistance=0, resistance=0.1
                        ):
    """
    计算逻辑：伤害 =
        攻击力 * (天赋倍率 * (1+基础伤害加成的乘算)) + 基础伤害加成的加算 + 【激化加成值-可选，不能和增幅冲突】
        * (1 + 各种增伤)
        * (1 + 暴击率*暴击伤害)
        * (反应基础倍率 * (1 + 精通提升 + 反应系数提高)) 【精通提升=(2.78 * em) / (em + 1400)】
        * (人等+100)/[(人等+100)+(1-穿防)*(1-减防+增幅)*(怪等+100)] 【即默认为190/[190+(1-穿防)*(1-减防)*190]】
        * 子(魔物抗性 - 减抗) 【f(x)=1-(x/2.0)[if x<0], =1-x[if 0≤x≤0.75], =1.0/(1+4*x)[if x>0.75]】
    :param atk: 指单一倍率角色的倍率所属性(如可莉是atk，芙宁娜是hp)
    :param talent: 如果是单一倍率，则按倍率算。如果是多个倍率，此处赋值1，让atk处先行计算完毕。
    :param em: 元素精通
```

在界面 1 中计算期望伤害的代码：

```

self.damage_expectation = dmg_cal.damagecal_detailedly(atk=self.atk_value, talent=self.talent_value, em=self.em_value,
catalyze_verify=self.catalyzeIf_value,
damage_catalyze_increased=self.DCI_value,
catalyze_type=self.catalyzeType_value,
added_basedamage=self.added_basedamage_value,
multiply_basedamage=self.multiply_basedamage_value,
db_increased=self.db_value,
cr=self.cr_value, cd=self.cd_value, cr100=0,
elemental_magnification=self.elemental_magnification,
increased_reaction_coefficient=self.IRC_value,
person_lever=self.person_lever_value,
hilichurl_level=self.hilichurl_level_value,
reduce_defenses=self.reduce_defenses_value,
ignore_defenses=self.ignore_defenses_value,
increase_defenses=self.increase_defenses_value,
reduce_resistance=self.reduce_resistance_value,
resistance=self.resistance_value
)

```

详细的代码见 [github](#)。注释我应该写的比较详细吧(

圣遗物部分

这是计算词条数的：

```

def cal_entryData(self):
    """
    [子函数]
    函数功能：计算词条数\n
    关联函数：cal_data_button
    参考资料：
    """
    hp      HP      atk      ATK      def      DEF      Charge      EM      CR      CD
    1档 209.13 4.08% 13.62 4.08% 16.20 5.10% 4.53% 16.32 2.72% 5.44%
    2档 239.00 4.66% 15.56 4.66% 18.52 5.83% 5.18% 18.65 3.11% 6.22%
    3档 268.88 5.25% 17.51 5.25% 20.83 6.56% 5.83% 20.98 3.50% 6.99%
    4档 298.75 5.83% 19.45 5.83% 23.15 7.29% 6.48% 23.31 3.89% 7.77%
    AVG 253.94 4.95% 16.54 4.95% 19.68 6.19% 5.51% 19.81 3.30% 6.60%
    """
    self.original_entryData = self.original_SmallHP_value / 253.94 + self.original_BigHP_value / 4.95 + \
self.original_SmallATK_value / 16.54 + self.original_BigATK_value / 4.95 + \
self.original_SmallDEF_value / 19.68 + self.original_BigDEF_value / 6.19 + \
self.original_Charge_value / 5.51 + self.original_EM_value / 19.81 + \
self.original_CritRate_value / 3.30 + self.original_CritDMG_value / 6.60
    self.originalEntryData_text.setText("词条数为: "+ "{:.4f}".format(self.original_entryData))

```

这是预测的：

```

def cal_entryData(self):
    """
    [子函数]
    函数功能：计算词条数\n
    关联函数：cal_data_button
    参考资料：
    """
    hp      HP      atk      ATK      def      DEF      Charge      EM      CR      CD
    1档 209.13 4.08% 13.62 4.08% 16.20 5.10% 4.53% 16.32 2.72% 5.44%
    2档 239.00 4.66% 15.56 4.66% 18.52 5.83% 5.18% 18.65 3.11% 6.22%
    3档 268.88 5.25% 17.51 5.25% 20.83 6.56% 5.83% 20.98 3.50% 6.99%
    4档 298.75 5.83% 19.45 5.83% 23.15 7.29% 6.48% 23.31 3.89% 7.77%
    AVG 253.94 4.95% 16.54 4.95% 19.68 6.19% 5.51% 19.81 3.30% 6.60%
    """
    self.original_entryData = self.original_SmallHP_value / 253.94 + self.original_BigHP_value / 4.95 + \
self.original_SmallATK_value / 16.54 + self.original_BigATK_value / 4.95 + \
self.original_SmallDEF_value / 19.68 + self.original_BigDEF_value / 6.19 + \
self.original_Charge_value / 5.51 + self.original_EM_value / 19.81 + \
self.original_CritRate_value / 3.30 + self.original_CritDMG_value / 6.60
    self.originalEntryData_text.setText("词条数为: "+ "{:.4f}".format(self.original_entryData))

```

预测的逻辑是：

```

# 将现有词条数转化为列表Current_entryData_list
Current_entryData = entryData
Current_entryData_list = [Current_entryData]

for _ in range(EnhanceTimes):
    Current_entryData_list = [xelem + addelem for xelem in Current_entryData_list for addelem in EnhanceList]
# print(str(artifactType) + "的 Current_entryData_list为:\n" + str(Current_entryData_list) + "\n[长度] + str(len(Current_entryData_list)))

```

每次迭代是 当前词条数加上每次强化的可能结果。

比较的逻辑是：

```
971 def CompareNumbers(self):
972     """
973     [子函数]
974     传入参数：列表，列表
975     比如现在新出的胚子是初始四从零开始强化，最后这个列表应该是16*16*16*16=1048576个元素。
976     以前的胚子是+12级的，就是16*16=256种强化结果。
977     然后比较1048576个结果和256种结果，得到超过的概率，1048576*256=268435456种情况中，现在的胚子比以前的强的概率。
978     :return: 概率
979     """
980     winwinwincount = 0
981     count = 0
982     for original_value in self.original_RubbishArtifact_list:
983         for current_value in self.current_RubbishArtifact_list:
984             count += 1
985             if current_value >= original_value:
986                 # 如果现在的圣遗物的预测词条数大于原来的圣遗物，则认为成功
987                 winwinwincount += 1
988     self.WinningProbability = 0
989     self.WinningProbability = winwinwincount/count # 现在的圣遗物赢了的概率
```

如果新胚子超过或等于原来胚子的词条数，就认为 win 了。

Part 5: 暂未实现的部分

```
找到5条TODO项 项位于4文件
▼ Genshin-Calculator 5 项
  ▼ damage_calculator.py 1 项
    (123, 10) TIP: todo 暂时只写90级激化的参考公式
  ▼ ui_cal.py 1 项
    (14, 3) # todo 做完后记得删除print相关代码
  ▼ ui_win1.py 2 项
    (164, 11) # todo 我靠忘了纯色队
    (507, 81) self.show_text_damage.setText('期望伤害:{self.damage_expectation}') # todo 未来考虑再做个append的窗口作为对比
  ▼ ui_win3.py 1 项
    (776, 11) # todo 如果初始3，需要EnhanceTimes=1 这个以后再做。并且要考虑初始3还要new一个新词条来预测词条。
```

1. 激化等级系数我只给了 90 级的
2. 纯色队的增幅应该恒为 1，我没写
3. 圣遗物预测不支持+0 级时的三词条圣遗物。（如果初始三词条但你已经强化到了+4 以上就没有影响）
4. 超导，扩散，碎冰，超载，感电，绽放，烈绽放，超绽放，燃烧均没写（其实我也不打算写 阿巴阿巴）