Component file

```
/*******************************************************************************
*
* McStas, neutron ray-tracing package
*         Copyright 1997-2002, All rights reserved
*         Risoe National Laboratory, Roskilde, Denmark
*         Institut Laue Langevin, Grenoble, France
*
* Component: Source_flat
*
* %I
* Written by: Kim Lefmann
* Date: October 30, 1997
* Modified by: KL, October 4, 2001
* Modified by: Emmanuel Farhi, October 30, 2001  Serious bug corrected
* Version: $Revision: 1.22 $
* Origin: Risoe
* Release: McStas 1.6
*
* A circular neutron source with flat energy spectrum and arbitrary flux
*
* %D
* The routine is a circular neutron source, which aims at a square target
* centered at the beam (in order to improve MC-acceptance rate).  The angular
* divergence is then given by the dimensions of the target.
* The neutron energy is uniformly distributed between E0-dE and E0+dE.
*
* Example: Source_flat(radius=0.1, dist=2, xw= 1, yh= 1, E0=14, dE=2)
*
* %P
* radius:  (m)   Radius of circle in (x,y,0) plane where neutrons
*                are generated.
* dist:    (m)   Distance to target along z axis
* xw:      (m)   Width(x) of target
* yh:      (m)   Height(y) of target
* E0:      (meV) Mean energy of neutrons.
* dE:      (meV) Energy spread of neutrons.
* Lambda0  (AA)  Mean wavelength of neutrons
* dLambda  (AA)  Wavelength spread of neutrons
* flux     (1/(s*cm**2*st)) Energy integrated flux
*
* %E
*******************************************************************************/

DEFINE COMPONENT Source_simple
DEFINITION PARAMETERS ()
SETTING PARAMETERS (radius, dist, xw, yh, E0=0, dE=0, Lambda0=0, dLambda=0, flux=1)
OUTPUT PARAMETERS ()
STATE PARAMETERS (x,y,z,vx,vy,vz,t,s1,s2,p)
DECLARE
%{
  double pmul, pdir;
%}
INITIALIZE
%{
  pmul=flux*PI*1e4*radius*radius/mcget_ncount();
%}
```

```
TRACE
%{
 double chi,E,Lambda,v,r, xf, yf, rf, dx, dy;

 t=0;
 z=0;

 chi=2*PI*rand01();                          /* Choose point on source */
 r=sqrt(rand01())*radius;                    /* with uniform distribution. */
 x=r*cos(chi);
 y=r*sin(chi);

 randvec_target_rect(&xf, &yf, &rf, &pdir,
        0, 0, dist, xw, yh, NOT_A_CURRENT_COMP);

 dx = xf-x;
 dy = yf-y;
 rf = sqrt(dx*dx+dy*dy+dist*dist);

 p = pdir*pmul;

 if(Lambda0==0) {
   E=E0+dE*randpm1();              /*  Choose from uniform distribution */
   v=sqrt(E)*SE2V;
 } else {
   Lambda=Lambda0+dLambda*randpm1();
   v = K2V*(2*PI/Lambda);
 }

 vz=v*dist/rf;
 vy=v*dy/rf;
 vx=v*dx/rf;
%}

MCDISPLAY
%{
  magnify("xy");
  circle("xy",0,0,0,radius);
%}

END
```

Written by developers and possibly you!