OpticalComponent< T, U > # std::unique ptr< Optical Surface< T, U > > surface # OpticalComponentType type + OpticalComponent(std ::unique_ptr< OpticalSurface < T, U > > surface, OpticalComponent Type type) + virtual ~OpticalComponent ()=default + const vec3< T > & getPosition () const + const vec3< T > & getNormal () const + OpticalSurface< T, U > * getSurfacePtr() + OpticalComponentType getType() const + virtual void handleLight (Ray< T, U > &ray, const vec3< T > &intersectionPoint)=0 Mirror< T, U > + Mirror(std::unique ptr< OpticalSurface < T, U > > surface) + virtual ~Mirror()=default + virtual void handleLight (Ray< T, U > &ray, const vec3< T > &intersectionPoint) override=0 # void reflect(Ray < T. U > &incidentRay, const vec3< T > &normal, const vec3< T > &intersectionPoint) ConvexMirror< T, U > + ConvexMirror(std::

unique_ptr< OpticalSurface < T, U > > surface) + void handleLight(Ray < T, U > &ray, const vec3< T > &intersectionPoint) override